VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra měřicí a řídicí techniky

Univerzální datalogger pro měření fyzikálních veličin s bezdrátovým přenosem dat

Universal Datalogger for Physical Quantities Measurement Including Data Wireless Transfer

2011 Martin Stařík

Poděkování

Děkuji vedoucímu své diplomové práce Ing. Radovanu Hájovskému, Ph.D. za podporu při tvorbě diplomové práce a vedení celého projektu. Dále bych chtěl poděkovat Ing. Luboši Urbanovi za pomoc při řešení problémů týkajících se programovatelného automatu TECOMAT FOXTROT a firmě Teco, a.s. za zapůjčení GSM terminálu v době opravy školního.

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 9. 5. 2011 .....

Stařík Martin

Abstrakt

Zadáním diplomové práce je využití univerzálního dataloggeru pro měření fyzikálních veličin s bezdrátovým přenosem dat. Datalogger je umístěn v měřicí stanici sestrojené podle návrhu zpracovaného v části diplomové práce Ing. Jiřího Žáčka z roku 2008 „Návrh a realizace autonomní stanice pro měření a archivaci dat“.
Původní program měřicí stanice byl touto diplomovou prací rozšířen o řízení měřicí stanice a archivaci dat pomoci GSM brány. Zdrojový kód je doplněn o dvě části - bezdrátovou komunikaci a archivaci dat. Bezdrátová komunikace je zajištěna GSM bránou GSM2-01, archivace dat je zajištěna uložením do paměťové karty. Aktuální naměřená data jsou pomocí příkazu prostřednictvím SMS zprávy přenesena do terminálu k dalšímu zpracování a zároveň archivována v paměti PLC. Vizualizace přijatých dat je zpracována v programovacím jazyku C#.

Abstract

The aim of this thesis is the use of a universal datalogger for the measurement of physical quantities with a wireless data transfer. The datalogger is set in a measuring station constructed on the basis of a design that can be found in one part of Ing. Jiri Zacek's diploma thesis called "The Design and Realization of an Autonomic System for Measuring and Data Storing".
This thesis had extended the original program of the measuring station to the control of the measuring station and data storing using the GSM gateway. Two parts had been added to the source code - wireless communication and data storing. The wireless communication is provided by the GSM gateway GSM2-01, where data storing is ensured by means of saving on a memory card. With the help of a command by way of a text message, current measured data are transferred to the terminal for further processing and also archived in the PLC memory. The visualization of the received data is processed in C #.

Klíčová slova

Archiv, Archivace, Autonomní režim,C#, CP-1004, Data, Datalogger, Ethernet, FOXTROT,GSM Modul, Konfigurace, Měření, Měřicí stanice, Mosaic, Programovatelný automat, Teco, TECOMAT, Vizualizace.

Key words

Archives, Archiving, Autonomous Mode,C#, CP-1004, Data, Datalogger, Ethernet, FOXTROT, GSM Modul, Configuration, Measurement, Measure station, Mosaic, Programmable Logic Controller, Teco, TECOMAT, Visualization.

Seznam použitých symbolů a zkratek

ADC	Analog Digital Konvertor. Analogově digitální převodník.
AI	Analog Input. Analogový vstup.
AO	Analog Output. Analogový výstup.
CAN	Controller Area Network. Výkonná sériová sběrnice pro tvorbu distribuovaných řídicích systémů.
CIB	Sběrnice osazená na PLC TECOMAT FOXTROT. Slouží k připojení jednotek systému Inels.
CPU	Central Processor Unit. Centrální procesorová jednotka.
DAC	Digital Analog Konvertor. Digitálně analogový převodník.
DI	Digital Input. Digitální vstup.
DIN	DIN lišta („U“ lišta). Konstrukční prvek k uchycení jističů a jiných zařízení v rozváděčových skříních apod.
DO	Digital Output. Digitální výstup.
ETHERNET	Síťová technologie používaná u lokálních sítí. Podporující různá média i šířky pásma.
EPSNET	Protokol EPSNET je protokol založený na standardu Profibus.
FBD	Function Block Diagram. Grafický jazyk, který znázorňuje signálové a datové toky mezi funkčními bloky. Je podobný hradlovým schématům.
Flash	nevolatilní (semipermanentní) paměť typu RAM (s náhodným přístupem), elektricky programovatelná.
GND	GrouND. Země, místo s nejnižším potenciálem.
GSM	Global System for Mobile communications. Síť pro mobilní komunikaci.
HW	Hardware.
IL	Instruction List. Textový programovací jazyk seznamu instrukcí. Připomíná asembler.
I/O	Input/Output. Vstup/výstup (vstupně-výstupní).
IP20	Označuje krytí přístroje.
IP ADRESA	32-bitová adresa přiřazena každému zařízení v síti, kde je použit TCP/IP protokol
IR	Infrared Radiation. Infračervené záření. IR přijímač = Přijímač IR signálu z dálkového ovládače.
LD	Ladder Diagram. Programovací jazyk příčkového diagramu. Je založen na grafické reprezentaci reléové logiky.
LED	Light Emitting Diode. Svítivá dioda.
LiON	Lithium Iontový akumulátor.
MMC	MultiMediaCard. Standard paměťové karty s technologií paměti flash.
MODBUS	Modbus je otevřený protokol pro vzájemnou komunikaci různých zařízení
OPC	OLE (Object Link and Embeding) for Process Control. Průmyslový standard pro komunikaci mezi zařízeními a softwarem.
PC	Personal Computer. Osobní počítač.
PLC	Programmable Logic Controller. Programovatelný logický automat.
POU	Program Organisation Unit. Programová organizační jednotka. Jedná se o nejmenší nezávislou část uživatelského programu.
Profibus DP	Komunikační standard používaný pro průmyslovou automatizaci na nejnižší úrovni řízení.
Profibus PA	Komunikační standard používaný pro průmyslovou automatizaci na nejnižší úrovni řízení. Jiná fyzická vrstva než Profibus DP.
RAM	Random-Access Memory. Typ paměti, která se používá především jako operační paměť počítačů.

RTC	Real Time Clock. Hodiny (obvod) reálného času.
RS-232	Typ rozhraní pro sériovou komunikaci.
RS-422	Typ rozhraní pro sériovou komunikaci.
RS-485	Typ rozhraní pro sériovou komunikaci.
SCADA	Supervisory Control and Data Acquisition. Sběr dat a řízení technologických procesů. SCADA systém = vizualizační systém.
SIM	Subscriber Identity Module. Účastnická identifikační karta, která slouží pro identifikaci účastníka v mobilní síti.
SMS	Short message service-Služba krátkých textových zpráv
SRAM	Static RAM. Tato paměť si uchovává informaci, dokud jí nevypneme napájení.
ST	Structured Text. Velmi výkonný vyšší programovací jazyk, který má kořeny ve známých jazycích Ada, Pascal a C.
SW	Software.
TCP/IP	Transmission Control Protocol/Internet Protocol. Sada síťových protokolů používaných v síti Internet, která poskytuje komunikaci v rámci vzájemně propojených sítí tvořených počítači s různou hardwarovou architekturou a různými operačními systémy.
TECO	Tesla Kolín. Firma zabývající se tvorbou řešení pro průmyslovou automatizaci.
TTL	Transistor Transistor Logic. Typ logiky, způsob provedení logiky digitálních integrovaných obvodů. TTL definuje napěťové úrovně signálu.
USB	Universal Serial Bus. Univerzální sériová sběrnice.
VAC	Střídavé napětí
VDC	Stejnosměrné napětí.

Obsah

1. Úvod

Sběr informací (dat) z průmyslových senzorů a čidel a jejích archivací zajišťují automatické systémy. Základním systémem jsou dataloggery, které dovedou vyčítat data ze senzorů a archivovat je. Dataloggery lze členit podle různých kritérií, například podle typu, počtu vstupů a výstupů, spolehlivosti atd.

V dnešní době se rozvíjí ovládání programovatelných automatů přes GSM bránu. Jde o zasílání SMS zpráv z mobilního zařízení do GSM brány umístěné u programovatelného automatu. Posíláním řídících SMS lze nastavovat zapínání a vypínání automatu, archivaci dat a jejich získávání. Tento druh ovládání se využívá například pro dálkové ovládání vytápění bytů, vzdálenou údržbu, diagnostiku a ovládání strojů.

V diplomové práci byla využita měřicí stanice, jejíž základní jednotkou je datalogger TECOMAT FOXTROT, který obsahuje 8 vstupů (4 digitálni, z toho jsou použity 2 jako čítače a 4 analogové). Datalogger je doplněn o GSM bránu, která zajišťuje komunikaci při bezdrátovém přenosu dat a paměťovou kartu, do níž se provádí archivace dat.

Diplomová práce řeší archivaci dat v měřicí stanici a její přenos pomocí GSM brány. Automat je v režimu autonomním, není tedy připojen k počítači a jeho ovládání je prováděno přes příkazy zasílané pomocí řídících SMS zpráv. Protože se jedná o asynchronní získávání naměřených dat, je podnětem pro archivaci a zasílání dat řídící SMS. Získávaná data, zasílaná prostřednictvím SMS zpráv jsou pomocí GSM terminálu připojeného k počítači ulkládána ve formě souboru typu CSV. Pro vizuálni kontrolu příjmáných dat je vytvořena aplikace v jazyce C#, která umožnuje data načíst a graficky zobrazit.

Podrobněji je tato problematika řešena v následujcích kapitolách diplomové práce.

2. Datalogger

- Co je to datalogger?

  Dataloggery jsou přístroje, které lze použít k ukládání dat. Zahrnují mnoho systémů sběru dat, jako jsou zásuvné karty do PC, nebo systémy se sériovou komunikací používající počítač k záznamu dat v reálném čase. Datalogger se považuje za samostatný přístroj, který umí číst různé typy elektrických signálů a uchovává data ve vnitřní paměti pro jejich pozdější přenos do počítače. Výhodou dataloggerů je jejich samostatná činnost nezávislá na počítači, na rozdíl od jiných přístrojů pro sběr dat. Tato řada zahrnuje jak jednoduché levné jednokanálové dataloggery s pevnou funkcí, tak i multifunkční programovatelné přístroje, které operují i se stovkami vstupů.

- Miniaturní dataloggery s jedním vstupem

  Miniaturní dataloggery s jedním vstupem jsou velmi levné a mají vždy jen určitý typ vstupu. Obvykle se využívají v dopravě.
  Typickou aplikací by bylo uložení dataloggeru pro teplotu přímo do zásilky potravinářských výrobků a ujistit se tak, zda teplota potravin nepřekročila dovolenou mez. Kromě teplotních dataloggerů jsou v nabídce i dataloggery s jinými různými vstupy.

- Vícekanálové dataloggery pro pevnou montáž

  Tyto dataloggery mají pevný počet vstupů a jsou obvykle určeny pro jeden typ vstupu.

- Přenosné vícekanálové dataloggery

  Přenosné vícekanálové dataloggery jsou používány jako stolní, nebo v laboratořích pro životní prostředí. Kromě ukládání do paměti mají některé modely vestavěnou tiskárničku pro okamžitý tisk kopie dat.

- Modulární dataloggery

  Modulární dataloggery lze konfigurovat a rozšiřovat pomocí zásuvných modulů. [1]

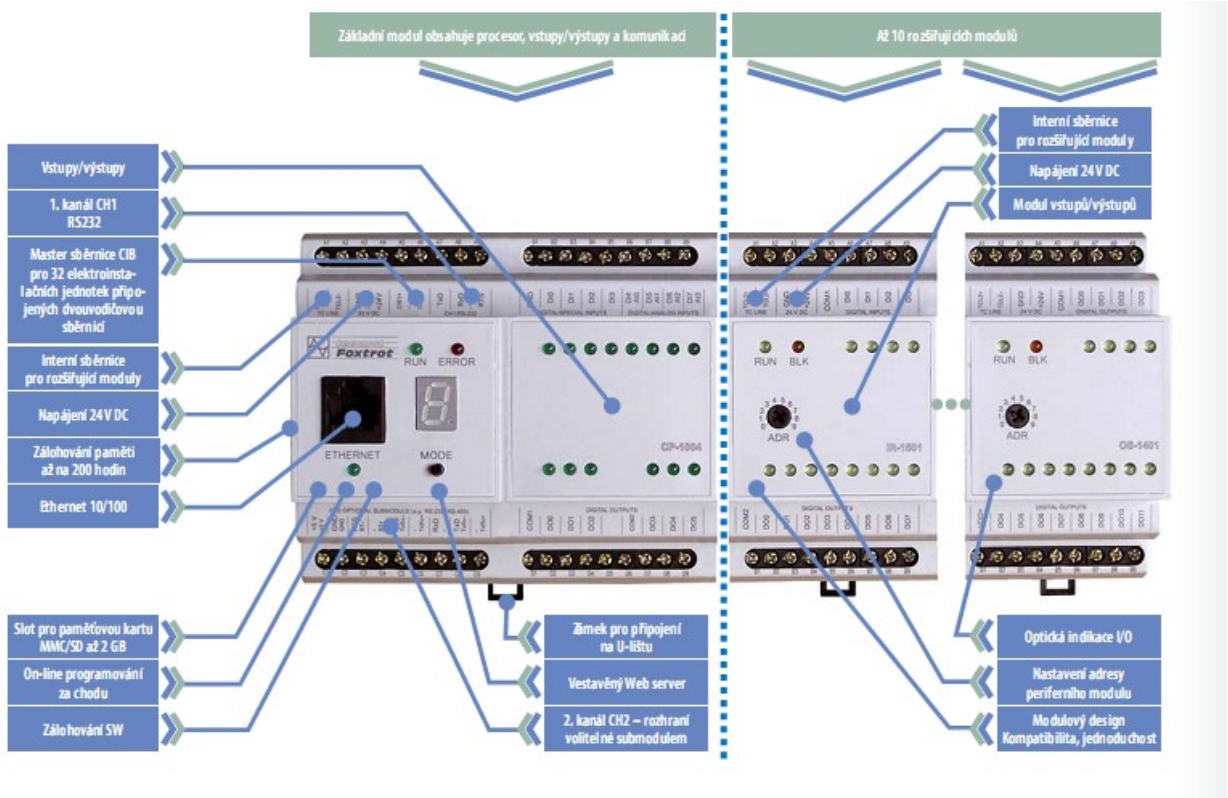
Obr. 2.1 Typy dataloggeru [1]

3. Základní pojmy PLC

3.1 Programovatelný automat (PLC)

PLC (Program Logic Controller) je volně programovatelný řídicí systém přizpůsobený pro řízení průmyslových a technologických procesů, mnohdy specializovaný na úlohy převážně logického typu (hlavně u starších typů a nejmenších systémů).

Menší systémy bývají řešeny jako kompaktní, větší jako modulární (stavebnice).



Obr. 3.1 PLC FOXTROT [3]

3.2 Přednosti PLC

- Hlavni předností PLC je možnost rychlé realizace systému.

Technické vybavení nemusí uživatel vyvíjet. Stačí navrhnout a včas objednat vhodnou sestavu modulů programovatelného automatu (konfiguraci) pro danou aplikaci, vytvořit projekt, vyvinout a odladit uživatelský program - a pak vše realizovat a uvést do chodu.

- Spolehlivost, odolnost, diagnostika

Technické vybavení programovatelných automatů je navrženo tak, že jsou extrémně spolehlivé i v těžkých průmyslových podmínkách, jsou odolné proti rušení i poruchám, vyznačují se robustností

a spolehlivostí. PLC bývají vybaveny také vnitřními diagnostickými funkcemi, které průběžně kontrolují činnost systému a včas zjistí případnou závadu, lokalizují ji, bezpečně ji ošetří a usnadní její odstranění.

3.3 Jiné počítače v automatizaci

- Osobní počítače

Někdy se setkáváme s přímým řízením technologických procesů standardním PC, mnohdy umístěným přímo v technologii. Toto řešení je přinejmenším riskantní a diskutabilní. Běžný počítač kategorie PC je produkt spotřební elektroniky a je konstruován pro povoz v prostředí domácností, laboratoří a kanceláří, kde obvykle funguje s vyhovující spolehlivostí. V průmyslových podmínkách mnohdy selhává (bývá málo spolehlivý, je citlivý na rušení a přepětí, nemá potřebnou životnost). Problémy vznikají již s pouhým připojením většího počtu vstupů a výstupů a s jejich odrušením. Proto se při přímém řízení strojů a technologií používají průmyslové počítače (IPC, IC), někdy jen v roli inteligentního operátorského panelu nebo komunikačního adaptéru (tzv. vestavné systémy).

- Jednočipové mikropočítače

Tak označujeme integrované obvody, které na jednom čipu sdružují přinejmenším mikroprocesor, generátor hodinového signálu, paměť i V/V brány v rozsahu umožňujícím alespoň v malé míře samostatnou činnost. Jsou vhodné všude tam, kde potřebujeme řešit automatický proces, který má menší počet vstupů a kde vystačíme s menší pamětí (Mikrokontrolér Intel 8051 má 4 KB vnitřní paměti programu). Tyto mikropočítače se využívají ve spotřební elektronice, ale také je nalezneme i v počítačích PC (klávesnice, zvuková karta).
Jednočipové mikropočítače neobsahují žádný operační systém. Programovat mikrokontroléry můžeme buď v základním jazyku, což je assembler. Nebo ve vyšších jazycích jako je například C.

- „soft PLC"

Tyto počítačové systémy představují zcela novou kategorii řídicích systémů. Mají uspořádání s centrálním počítačem a se souborem distribuovaných pasivních modulů, které jsou spojeny rychlou a spolehlivou průmyslovou sběrnicí. Inteligence systému je soustředěna v počítači.

Komunikační funkce a funkce, související se spolehlivým fungováním v reálném čase a se zajištěním dostatečně rychlé odezvy na vnější události, zajišťuje buď samotný počítač (PC, IPC) se standardním HW (a obvykle se specializovaným nebo upraveným operačním systémem), nebo se specializovaným „PLC koprocesorem" (zásuvným modulem pro PC s vlastním procesorem, spolehlivým operačním systémem reálného času a s vlastním programovým vybavením), případně samostatný počítačový modul, sériově komunikující s PC. Centrální počítač realizuje všechny řídicí funkce PLC (uživatelský program PLC a systémové služby PLC) a funkce vývojového systému (zadání a překlad programu, jeho simulaci, podporu při ladění programu, při uvádění řízeného procesu do provozu a při jeho diagnostice).

Systémy Soft PLC nabízí tradiční programovací jazyky pro PLC (obvykle podle standardu IEC 1131-3), nabízí ale navíc i standardní služby operačního systému a standardních programových produktů (např. komunikační a archivační funkce, databázové a výpočetní operace, programování v jazycích Pascal, C, tabulkové a textové procesory, výkonné simulační a výpočetní programy apod.).

Výhodou Soft PLC je možnost soustředit (integrovat) v jednom systému funkce PLC a jeho vývojového prostředí spolu s operátorským rozhraním, s vizualizací a archivací technologických dat, s optimalizačními, expertními a diagnostickými funkcemi.

Soft PLC se prozatím nevyužívají ve velké míře. Jejich rozšíření brání především vysoká cena (technického i programového vybavení). Potencionální koncoví uživatelé stále požadují standardní programovatelné automaty.

Principiální nevýhodou Soft PLC ale zůstává centralizace funkcí do jednoho počítače a z něj plynoucí zranitelnost a závislost na počítači. Řízení jednotlivých strojů a nepříliš složitých technologických procesů, kde není zdůvodněno použití počítače, proto ještě dlouho zůstane výlučnou doménou pro aplikace tradičních PLC.

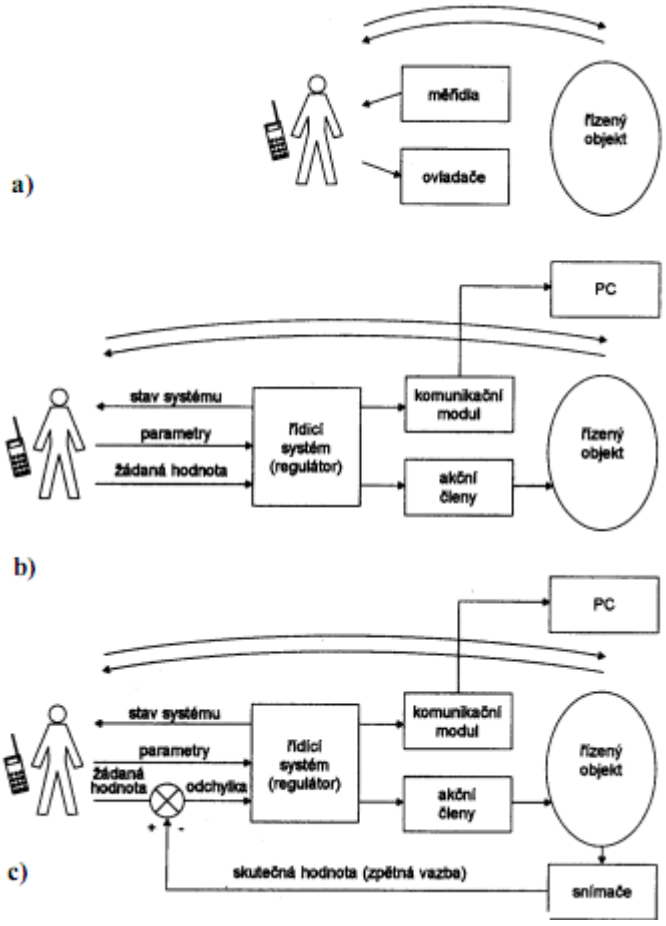
Prostor pro nasazování Soft PLC lze očekávat v případech náročného řízení rozsáhlých procesů, zejména v případech, kdy jsou nutné velmi náročné výpočetní algoritmy a kde je požadováno zpracování a archivace velkých objemů dat.

3.4 PLC a řízení

Začlenění PLC do systému řízení je znázorněno na obr. 3.4a. Při ručním řízení vykonává všechny operace člověk. Při přímém (dopředném) řízení (obr.3.4b) působí PLC na řízený objekt jednosměrně, jen jej ovládá a nekontroluje dosažený stav. Mezi systémem a řízeným objektem jsou zařazeny jen akční členy. Při zpětnovazebním řízení (obr. 3.4c) získává řídicí systém zpětnou informaci o stavu řízeného objektu (realizuje zpětnou vazbu, uzavírá zpětnovazební smyčku). Porovnává požadovaný stav se skutečným, a podle zjištěné odchylky upravuje své akční zásahy tak, aby dosáhl požadovaného stavu (nebo se mu alespoň co možná nejvíce přiblížil).

Zpětnovazební řízení je typické pro regulační úlohy. Při použití PLC to znamená, že zadání žádané hodnoty je provedeno v číslicové formě, s číselnou informací systém operuje i při zpracování skutečné hodnoty a odchylky, ale i při výpočtech pomocných veličin potřebných k realizaci regulačního algoritmu. Řízený objekt je proto třeba doplnit o potřebné snímače pro měření stavu sledovaných veličin (např. teploty, hladiny, polohy, nebo tlaku). Za zpětnovazební řízení ale můžeme považovat i logické řízení, při kterém na objekt působíme jen dvouhodnotovými povely typu „vypni - zapni" a zpracováváme i zpětnovazební informace dvouhodnotového charakteru ve významu hlášení o vykonání povelu, nebo překročení povolených hodnot (např. informace typu: „hladina nízká", „hladina dosažena", „hladina překročena", „nádrž prázdná", „nádrž přeplněna" apod.). Pro oba případy je navíc naznačena komunikační vazba řídicího systému k nadřízenému počítačovému systému (např. pro monitorování procesu). Ponechána je i účast člověka na řízení procesu, protože i v automatizovaných procesech bývá jeho přítomnost nezbytná (alespoň občasná, pro kontrolu a seřízení).

Situace na obr. 3.4.1 je zjednodušena. V praxi je běžná kombinace všech tří způsobů řízení. Mnohdy se i při ručním řízení uplatňuje řídicí systém, nejčastěji PLC. Obvykle je nezbytný už jen k obsluze, ke komunikaci s operátorským panelem, ke zpracování povelů operátora, k vyhodnocení stavů stroje a k jejich zobrazení, jako prostředník mezi povely operátora a mezi jednotlivými akcemi pro řízení stroje, pro měření a pro zpracování měřené informace, pro logické ochrany stroje apod. Mnohdy je při řízení stroje nutné zajistit složité posloupnosti dílčích akci, zajistit koordinaci povelů pro pohony s jinými akčními zásahy, jejich a kontrolu apod. [2]



Obr. 3.4.1 Schéma způsobu řízení [2]

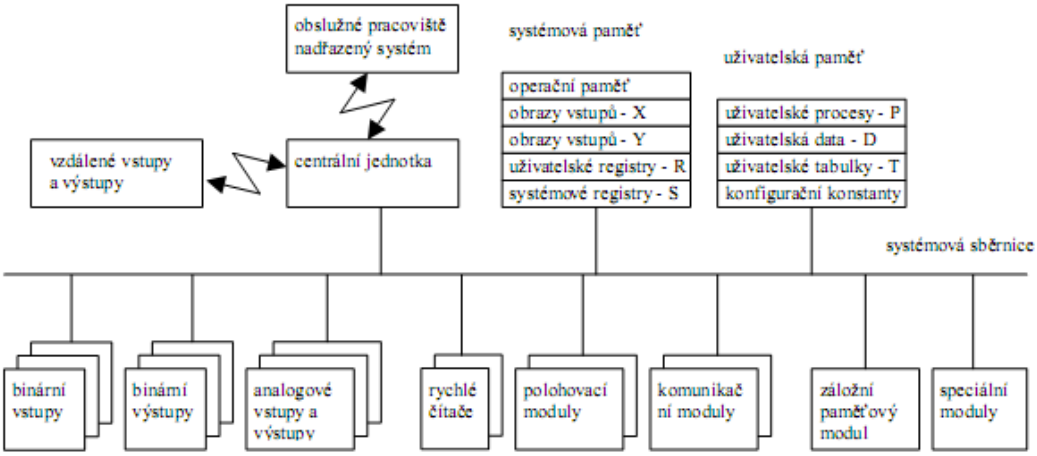
4. Struktura PLC

Na našem trhu je možno se nejčastěji setkat s programovatelnými automaty těchto nejvýznamnějších světových výrobců (řazeno abecedně): ABB, Allen-Bradley, B+R, Eberle, Festo, GE, H+B, Idec, Klockner Moeller, Matsushita, Mitsubishi, Omron, Saia, Siemens, Schneider Group a českého výrobce Teco, a.s. V detailech se jednotlivé třídy systémů a jejich představitelé liší, způsoby použití a aplikační možnosti jsou však srovnatelné.

4.1. Volba konfigurace PLC

Blokové schéma struktury typického programovatelného automatu, tak jak je uvedeno na obr. 4.1.1, je třeba chápat jen jako možnou konfiguraci pro náročnou aplikaci. Skutečnou sestavu volí uživatel tak, aby co nejlépe přizpůsobil PLC požadavkům řešené úlohy. V konkrétním případě mohou některé typy modulů chybět, jiné se mnohonásobně opakují. V krajním případě může být PLC vystavěn jako čistě binární (logický) systém, s výhradním použitím dvouhodnotových vstupů a výstupů, nebo naopak jako výhradně analogový (regulátor, měřicí nebo monitorovací systém). Mohou být zdůvodněné i sestavy čistě vstupní, kdy je PLC degradován na systém pro měření a předzpracování dat. PLC může například vyhodnocovat soubor analogových a binárních snímačů z monitorované technologie, analyzovat je, nebo předávat nadřízenému PC. Může se specializovat jen na čítání impulzů z vysílacích elektroměrů, z impulzních průtokoměrů plynu, teplé a studené vody, nebo z měřičů spotřebovaného tepla. Obdobně může být PLC v roli čistě výstupního systému, například jako ovládač sviticích nebo padáčkových segmentových zobrazovačů, souboru pohonů nebo souboru elektrických spotřebičů a jiných akčních členů.

Existují i aplikace PLC bez fyzických vstupů a výstupů, kdy PLC funguje jen jako inteligentní a programovatelný komunikační adaptér (pro připojení „cizího systému" do sítě PLC, pro připojení operátorských panelů, snímačů čárového kódu a jiných identifikačních prvků, vážících zařízení, jako ovládač tiskárny, rádiového nebo telefonního modemu pro dálkové ovládání, jako inteligentní převodník komunikačních rozhraní a adaptér mezi protokoly různých průmyslových sběrnic).



Obr. 4.1.1 Blokové schéma vnitřní struktury programovatelného automatu [2]

4.2. Mikro PLC

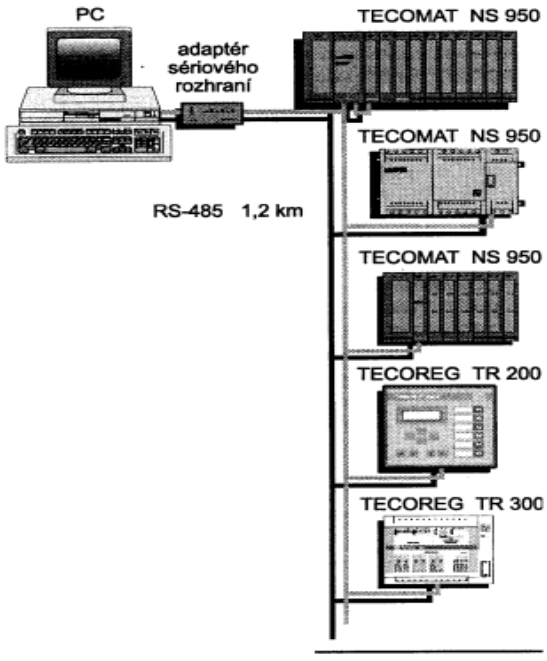
Nejmenší a nejlevnější kompaktní PLC systémy (mikro PLC) nabízejí uživateli pevnou sestavu vstupů a výstupů, obvykle jen binárních, například 6 binárních vstupů/6 binárních výstupů pro nejmenší systém. Uživatel se v tomto případě může rozhodnout pro jeden typ systému, který již nemůže dodatečně rozšiřovat. Svým kompaktním provedením, malými rozměry a nízkou cenou (v jednotkách tisíc Kč) se mikro PLC řadí do kategorie „spotřebního materiálu". Jejich funkční a programátorský komfort je obvykle redukován na nezbytné minimum, komunikační možnosti mnohdy chybějí. Typickým použitím programovatelných automatů nejnižší kategorie (mikro PLC) je realizace logické výbavy jednoduchých strojů a mechanismů, která se tradičně řešila pevnou reléovou logikou.

4.3. Kompaktní PLC

Ostatní PLC v kompaktním provedení nabízejí určitou, i když omezenou variabilnost ve volbě konfigurace. Uživatel může k základnímu modulu připojit jeden nebo několik přídavných modulů z omezeného sortimentu s pevnou kombinací vstupů a výstupů, např. modul s 8 binárními vstupy a 8 binárními výstupy (tranzistorovými nebo reléovými), modul rychlých čítačů, analogový vstupní nebo výstupní modul, modul regulátoru apod. Některé kompaktní systémy se navíc vyznačují ještě vnitřní modulárností, kdy konfiguraci základního modulu lze sestavit osazením základní desky násuvnými moduly vhodného typu („piggyback").

4.4. Modulární PLC

Nesrovnatelně větší volnost ve volbě konfigurace poskytují modulární programovatelné automaty. Do různých variant plochého zadního rámu lze zasouvat libovolné moduly (typicky v počtu 4, 6, 8 a 11 modulů). U některých variant může být jeden systém tvořen několika rámy (základní a rozšiřovací moduly). Rozšiřovací moduly mohou být připojeny na vzdálenosti stovek metrů. Místo rozšiřujících modulů mohou být připojeny podsystémy tvořené kteroukoliv z variant. Tak lze vytvářet různě strukturované distribuované systémy - např. podle obr. 4.4.1. [2]



Obr. 4.4.1 Ukázka distribuovaného systému vytvořeného pomocí PLC fy TECO, a.s. [2]

4.5 Hlavní prvky programovatelného automatu

- Binárními vstupy se připojují tlačítka, přepínače, koncové spínače a jiné snímače s dvouhodnotovým charakterem signálu (např. dvouhodnotové snímače tlaku, teploty nebo hladiny).

- Binární výstupy jsou určeny k buzení cívek relé, stykačů, elektromagnetických spojek, pneumatických a hydraulických převodníku, k ovládání signálek, ale i ke stupňovitému řízení pohonu a frekvenčních měničů. Analogové vstupní a výstupní moduly zprostředkovávají kontakt programovatelného automatu s prostředím.

- Analogovými vstupy se připojují snímače teploty (obvykle odporové, polovodičové nebo termočlánky), snímače tlaku, vlhkosti, hladiny, a také většina inteligentních přístrojů s analogovými výstupy.

- Analogové výstupy pomáhají ovládat spojité servopohony a frekvenční měniče, ale také ručkové měřicí přístroje a jiné spojitě ovládané akční členy.

- Centrální procesorová jednotka (CPU) je hlavní částí programovatelného automatu. Realizuje soubor instrukcí a systémových služeb, zajišťuje i základní komunikační funkce s vlastními i vzdálenými moduly, s nadřízeným systémem a s programovacím přístrojem. Obsahuje mikroprocesor a řadič, zaměřený na rychlé provádění instrukcí.

- V paměti jsou uloženy uživatelské registry, čítače a časovače, komunikační, časové a jiné systémové proměnné. Paměť slouží také k uložení uživatelského programu. Na rozdíl od počítače si PLC při poruše řídicího systému musí zapamatovat poslední stav, od něhož po obnovení funkce pokračuje dál v činnosti, což klade nároky na velký objem paměti.

5. TECOMAT PLC

5.1 Přehled výrobků

Teco, a.s., je výrobcem programovatelných řídicích systémů a příslušenství pro průmyslovou automatizaci.

- Programovatelné řídicí systémy TECOMAT (PLC)
- Programovatelné regulátory TECOREG (DDC)
- Inteligentní elektroinstalační systém INELS
- Průmyslové panelové počítače TEMPO
- Textové operátorské panely
- Grafické operátorské panely
- Svorkovnicové moduly
- Napájecí zdroje
- Vestavné moduly
- Signalizační zařízení

5.2 TECOMAT – přehled modulů

Programovatelné automaty TECOMAT jsou určeny pro řízení nejrůznějších technologií, strojů a linek ve všech průmyslových oborech, pro nasazení v energetice, dopravě apod. Svým rozsahem pokrývají aplikace od jednotek vstupů a výstupů (malé kompaktní systémy TC400) přes větší kompaktní systémy TC500, TC600 a TC650, až po velké aplikace, pro které jsou určeny modulární systémy TC700 (tab. 5.2.1.). Novinkou jsou malé modulární PLC řady TECOMAT FOXTROT s rychlým Ethernetem a vestavěným webserverem.

Typová řada	Binární I/O	Analogové I/O	Komunikační kanály	Paměť programu
TC400	6/4	2/-	2 serial	32 kB
TC500	20/20	4/4	2 serial	32 kB
TC600	48/44	24/8	3 serial	32 kB
TC650	48/44	24/8	3 serial, 1 Ethernet	64 + 64 kB
FOXTROT	Až 134 DI/DO	až 80AI / 20AO	2×serial,1×CIB,1×Ethernet	192 + 64 kB
TC700	přes 6500	až 3300/700	až 10 serial, 2 Ethernet, 1 USB	192 + 64 kB

Tab. 5.2.1 Přehled modulu TECOMAT[3]

Hlavními rysy modulárního systému TECOMAT FOXTROT v HW jsou:

- nový 32bitový RISC procesor s frekvencí 166MHz,
- 192 kB zálohované paměti pro program a 64 kB pro tabulky,
- paměť rozšiřitelná kartami standardu SD/MMC o velikosti až 2 GB, případně SDHC až 16 GB,
- integrované rozhraní Ethernet 10/100 Mbit/s na konektoru RJ45,
- design rozměrově kompatibilní se standardizovanými moduly jističů,
- rozšiřitelnost počtů vstupů/výstupů systémovou sběrnicí označenou TCL2,
- kombinované analogové/diskrétní a čítačové/ diskrétní vstupy na základním modulu,

- integrovaná dvouvodičová instalační sběrnice CIB (Common Installation Bus) s volnou topologií pro připojení inteligentních elektroinstalačních prvků,
- možnost spojení s operátorským panelem ID-14 v kompaktní celek,
- varianta s integrovaným displejem 4×20 znaků a 6 funkčními tlačitky.[3]

Typ	Specifikace	Objednací číslo
CP-1004	základní modul, 8×DI 24 V DC (4 vstupy i AI 10 bit, 0-10 V), 6×RO 230 V AC/3A, 1×Ethernet, 2×SCH, 1×CIB	TXN 110 04
CP-1005	základní modul, 6×DI/AI, 2×AO, 6×RO 230V/3A, 1×Ethernet, 2×SCH, 1×CIB	TXN 110 05
CP-1014	základní modul, 8×DI 24 V DC (4 vstupy i AI 10 bit, 0-10 V), 6×RO 230 V AC/3A, 1×Ethernet, 2×SCH, 1×CIB, LCD 4×20 znaků, 6 tlačítek	TXN 110 14
CP-1015	základní modul, 6×DI/AI, 2×AO, 6×RO 230V/3A, 1×Ethernet, 2×SCH, 1×CIB, LCD 4×20 znaků, 6 tlačítek	TXN 110 15
CP-1016 *	základní modul, 6×DI/AI (RTD),7×DI/AI (0-20mA,RTD), 1×IMP (průtokoměr),1×DI 230VAC (HDO), 2×AO 0-10V, 2×AO 0-10V, 2×DO 230VAC SSR, 10×RO 230V/3A, 1×Ethernet, 2×SCH, 1×CIB,1×RF, LCD 4×20 znaků, 7 tlačítek	TXN 110 16
IB-1301	binární rozšiřovací modul, 12×DI 24 V, běžné vstupy	TXN 113 01
OS-1401	binární rozšiřovací modul, 12×DO 24 V DC/0,5 A, se společnou svorkou	TXN 114 01
IR-1501	binární rozšiřovací modul, 4×DI 24 V AC/DC, 8×RO 230V AC/2A	TXN 115 01
IT-1601	analogový rozšiřovací modul, 8×AI 16bit, 0-20 mA, 0-10 V, Ni1000, Pt1000, Pt100, 2×AO 8 bit, 0-10 V	TXN 116 01
IT-1602	analogový rozšiřovací modul, 8×AI 16bit, 0-1 V, termočlánky J, K, R, S, T, N, 2×AO 8 bit, 0-10 V	TXN 116 02
KB-0552	modul optického propojení sběrnice	TXN 105 52
IS-0601	modul pro připojení ostrovních ventilů FESTO	TXN 106 01

Tab.5.2.2 Přehled modulů TECOMAT FOXTROT[3]

5.3 TECOMAT FOXTROT

5.3.1 Popis

Programovatelné automaty TECOMAT FOXTROT představují malé kompaktní automaty s možností modulárního rozšíření. Spojují tak výhody kompaktních automatů co do velikosti, a modulárních co do rozšiřitelností a variability.
Jsou určeny pro řízení technologií v nejrůznějších oblastech průmyslu i v jiných odvětvích. Jednotlivé moduly systému jsou uzavřeny v plastových ochranných pouzdrech, které se montují na U lištu ČSN EN 50022. Díky tomu lze s nimi manipulovat bez nebezpečí poškození citlivých CMOS součástek.

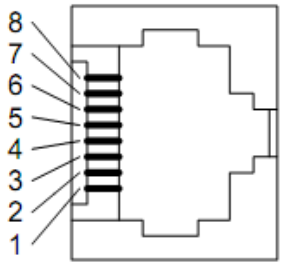
Obr. 5.3.1.1 TECOMAT FOXTROT[4]

5.3.2 Komunikace

Datové komunikace mezi PLC a nadřízenými PC, mezi několika PLC, nebo mezi PLC a ostatními zařízeními jsou obvykle realizovány sériovými přenosy. Systémy FOXTROT podporují základní přenosy pomocí sítí Ethernet nebo průmyslové sítě EPSNET. Jeden asynchronní sériový kanál je pevně osazen rozhraním RS-232, druhý je volitelně osazen různými typy fyzických rozhraní podle volby zákazníka (RS-232, RS-485, RS-422). Na jedné úrovni sítě EPSNET může být při použití rozhraní RS-485 až 32 účastníků a délka sériové linky až 1200 m. Volitelně jsou podporovány i jiné průmyslové protokoly a sběrnice, např. MODBUS, PROFIBUS DP, CAN, apod. Případně je možná asynchronní komunikace univerzálními přenosovými kanály, ovládanými přímo z uživatelského programu. Všechny centrální jednotky jsou vybaveny rozhraním Ethernet 10/100Mb umožňujícím provozovat současně více logických spojení.

5.3.2.1 Rozhraní Ethernet

Základní moduly jsou osazeny rozhraním Ethernet 10/100Mbit. Rozhraní Ethernet je osazeno konektorem RJ-45 se standardním rozmístěním signálů. Konektor je připraven pro použití běžných UTP patch kabelů. Rozhraní je zkonstruováno tak, že umožňuje použití jak přímých, tak křížených kabelů.



Pin	Signál	Barva vodiče
8	nepoužitý	hnědý
7	nepoužitý	bílý / hnědý
6	RD– nebo TD–	zelený nebo oranžový
5	nepoužitý	bílý / modrý
4	nepoužitý	modrý
3	RD+ nebo TD+	bílý / zelený nebo bílý / oranžový
2	TD– nebo RD–	oranžový nebo zelený
1	TD+ nebo RD+	bílý / oranžový nebo bílý / zelený

Obr. 5.3.2.1.1 Rozhraní Ethernet [4]

5.3.3 Programovací přístroj

Jako programovací přístroj lze použít počítač PC. Konfiguraci počítače je nutné zvolit
podle požadavků programového vybavení (Mosaic, Reliance…). TECOMAT FOXTROT nabízí řadu užitečných systémových služeb, které zjednodušují a zpříjemňují programování. Příkladem může být pestrá škála časových údajů, zveřejněné aktuální datum a čas, nebo systémová podpora pro ošetřování stavů při zapnutí napájení PLC.

5.3.4 Programování

Veškeré systémy TECOMAT lze programovat z vývojového prostředí Mosaic. Společným programovacím jazykem je firemní mnemokód (jazyk typu IL), který zajišťuje kompatibilitu programování a přenositelnost kódu mezi jednotlivými systémy TECOMAT. U nových typů PLC TECOMAT založených na 32bitových procesorech je možné psát uživatelský program v jazyce strukturovaného textu podle mezinárodní normy IEC EN 61131-3, což přináší řadu výhod - mj. zpřehledňuje a zefektivňuje programování a zároveň je možné využít podpůrné nástroje, které nejsou u starších typů CPU podporovány.

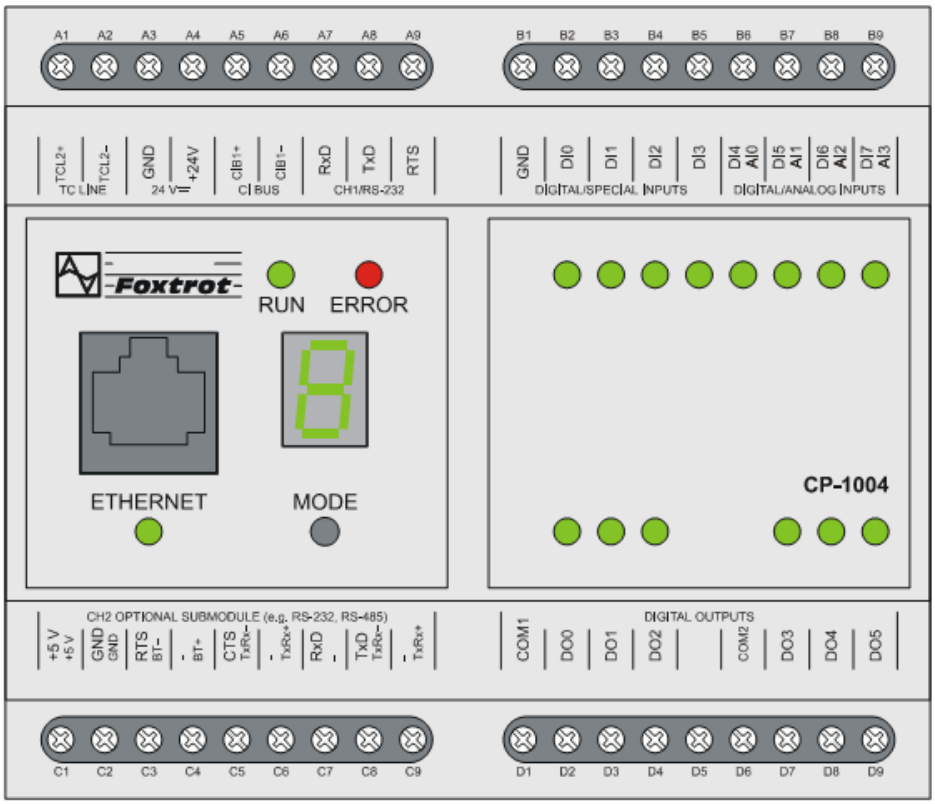
5.4 TECOMAT FOXTROT CP-1004

5.4.1 Popis

Centrální jednotka řady K
4 binární vstupy 24 V využitelné jako vstupy čítačů
4 volitelné vstupy - binární 24 V / analogové 0 - 10 V (10 bitů)
6 reléových výstupů 250 V
2 sériové kanály (CH1 - RS-232, CH2 - volitelné rozhraní)
1 rozhraní Ethernet 10/100 Mb
1 linka sběrnice TCL2 pro připojení periferií
1 linka sběrnice CIB
Slot paměťové karty SD / MMC
Možnost osazení submodulu s binárními vstupy a výstupy

5.4.2 Periferní část modulů CP-1004

Periferní část modulů CP-1004 tvoří deska IR-1057 (starší provedení IR-1055) obsahující 8 víceúčelových vstupů a 6 reléových výstupů. První čtyři vstupy DI0 - DI3 mohou být použity jako běžné binární vstupy nebo jako vstupy pro čítače. Další čtyři vstupy DI4 - DI7 mohou být použity jako běžné binární vstupy nebo jako analogové vstupy AI0 - AI3. Pod jménem IR-1057 (resp. IR-1055) se hlásí na systémové sběrnici procesor obsluhující tyto vstupy a výstupy.



Obr. 5.4.2.1 Základní modul CP-1004[4]

- Binární vstupy

Binární vstupy slouží k připojení dvoustavových signálů řízeného objektu k PLC. Základní modul CP-1004 obsahuje 8 binárních vstupů DI0 - DI7. Vstupy nejsou galvanicky odděleny od vnitřních obvodů PLC. Vybuzení (sepnutí) vstupu je signalizováno rozsvícením příslušné LED diody. Všechny vstupy mají jednu společnou svorku minus. Vstupy DI0 - DI3 lze použít jako vstupy pro čítače, vstupy DI4 - DI7 lze použít jako analogové vstupy AI0 - AI3. I v případě využití jako vstupy pro čítače jsou vstupy DI0 - DI3 současně použitelné jako binární. Vstupy DI4 - DI7 pracují jako binární pouze tehdy, pokud nejsou použity pro analogové měření (platí pro každý vstup nezávisle na ostatních). Vstupy DI0 - DI3 umožňují zapnout funkci zachytávání krátkých pulzů. Tato funkce prodlužuje zvolenou úroveň vstupního signálu až do otočky PLC. Tak zajistíme, že nepřijdeme o jednotlivý pulz na vstupu kratší než doba cyklu PLC.

- Reléové výstupy

Reléové výstupy slouží k ovládání dvoustavových akčních a signalizačních prvků řízeného objektu, napájených střídavým nebo stejnosměrným napětím až do 250 V. Výstupy jsou realizovány spínacím beznapěťovým kontaktem relé vyvedeným ve skupině s jednou společnou svorkou. Základní modul CP-1004 obsahuje 6 reléových výstupů DO0 - DO5 organizovaných ve dvou skupinách po třech výstupech se společnou svorkou. Výstupy jsou galvanicky odděleny jak od vnitřních obvodů PLC, tak obě skupiny mezi sebou. Vybuzení (sepnutí) výstupu je signalizováno rozsvícením příslušné LED diody.

- Analogové vstupy

Analogové vstupy slouží k připojení analogových signálů řízeného objektu k PLC. Základní modul CP-1004 obsahuje 4 analogové vstupy AI0 - AI3, které jsou fyzicky shodné s binárními vstupy DI4 - DI7. Vstupy nejsou galvanicky odděleny od vnitřních obvodů PLC. Všechny vstupy mají jednu společnou svorku minus. Vstupy DI0 - DI3 lze použít jako vstupy pro čítače, vstupy DI4 - DI7 lze použít jako analogové vstupy AI0 - AI3. Pokud jednotlivý vstup ze skupiny DI4 - DI7 / AI0 - AI3 není použit pro analogové měření, pracuje jako binární vstup.

- Čítače

Binární vstupy DI0 - DI3 lze použít jako vstupy pro čítače. K dispozici jsou dva objekty čítačů, které mohou pracovat v několika režimech (jednosměrný čítač, obousměrný čítač, základní IRC). Každý objekt čítače standardně využívá dva vstupy. První objekt čítače navíc umožňuje i režimy, které používají všechny čtyři vstupy (čítač a IRC s nulováním a zachytáváním, měření délky pulzu, měření periody a fázového posunu). V tom případě je druhý objekt čítače vypnut. I při použití pro tyto alternativní funkce jsou vstupy DI0 - DI3 současně použitelné jako běžné binární.[4]

6. Vývojové prostředí

6.1 Vývojové prostředí PLC TECOMAT

- **SoftPLC for Windows**
  SoftPLC je produkt běžící pod Windows 2000/XP, který poskytuje uživatelům funkce a možnosti PLC řady TC700 pro vývoj a testování PLC aplikací bez nutnosti vlastnit reálný hardware.

- **Merkur**
  Merkur je grafické vývojové prostředí určené pouze pro regulátory TECOREG. K programování regulátorů se používá jazyk typu FBD.

- **Epos for Windows**
  Epos for Windows je starší verze vývojového prostředí pro PLC TECOMAT, používající textový programovací jazyk JLR (strukturovaný text).

- **xPRO**
  xPRO je vývojové prostředí pro programování PLC TECOMAT pod operačním systémem MS DOS, ale lze jej spustit i pod Windows 95/98.

- **Quick II**
  Quick II je jednoduchý program pro tvorbu aplikací s programovatelnými relé SMART. Program je složen z funkčních bloků.

- **MOSAIC**
  Vývojové prostředí pro PLC TECOMAT a regulátory TECOREG podle normy IEC 61131-3

6.2 MOSAIC

MOSAIC je integrované vývojové prostředí, které umožňuje vytvářet aplikační programy jak pro PLC TECOMAT, tak pro regulátory TECOREG. Prostředí umožňuje programování v jazyce instrukcí (mnemokód), systémy s 32 bitovými procesory (TECOMAT TC650 a TC700), lze programovat také v jazycích podle IEC EN 61131-3 (IL, ST, LD, FBD). Součástí prostředí MOSAIC je i řada nástrojů usnadňujících vývoj a ladění aplikací. Prostředí zachovává kompatibilitu se starším MS-DOS prostředím xPRO a umožňuje pracovat se zdrojovými programy vytvořenými v tomto prostředí.

6.2.1 Vlastnosti

- Pracuje pod Windows 2000/XP/.

- Podporuje programování v jazycích ST (strukturovaný text), IL (jazyk instrukcí), LD (reléové schéma) a FBD (funkční bloky) podle normy IEC 61131-3.

- IEC manažer pro grafickou deklaraci všech prvků programu PLC - datových typů, proměnných, funkcí, funkčních bloků i programových jednotek; možnost deklarace vlastních knihoven.

- Inspektor POU - nástroj pro ladění programu PLC, sleduje a zobrazuje stav vybraných proměnných, umožňuje používat ladicí body.

- Simulátor PLC - dovoluje ladit programy bez nutnosti připojení reálného hardwaru, simulovat lze všechny typy PLC TECOMAT a TECOREG; k simulátoru lze připojit i vizualizační software RELIANCE a ladit celou aplikaci na jednom PC.

- PanelMaker - nástroj na tvorbu dialogů pro operátorské panely ID-07, ID-08 a PLC řady TC500 a TR200; program pro panel je součástí programu pro PLC.

- PanelSim - simulátor operátorských panelů dovoluje zkoušet dialogy vytvořené PanelMakerem bez připojení skutečného panelu, funguje jak s reálným, tak i simulovaným PLC.

- PIDMaker - nástroj pro ladění a návrh PID regulátorů; nabízí interaktivní náhled na průběh regulace, usnadňuje správné nastavení parametrů regulátoru a generuje programový kód. Součástí je simulace jednoduchých soustav do třetího řádu s dopravním zpožděním.

- GraphMaker - nástroj pro podporu ladění a diagnostiku řízeného systému umožňuje zobrazení průběhů vybraných proměnných offline i v reálném čase. Obsahuje dva sledovací kurzory s nastavitelnou periodou vzorkování, umožňuje ukládání dat na disk i export do DB programů. Zahrnuje také funkce digitálního osciloskopu (16 kanálů) a logického analyzátoru.

- Softwarová konfigurace PLC - konfigurační nástroj umožňující výběr typu PLC a definici konkrétní sestavy včetně nastavení parametrů jednotlivých modulů. Rovněž umožňuje načíst aktuální konfiguraci z připojeného PLC.

- Definice sítě PLC - nástroj umožňuje grafickou formou vytvořit vazby mezi PLC v rámci projektu, definovat připojení operátorských panelů nebo externích zařízení.

- Projektový manažer - komfortní správa projektu, archivace a zálohování projektu.

- hypertextová a kontextová nápověda, zahrnuje kompletní dokumentaci k systémům TECOMAT a TECOREG ve formátu pdf.

6.2.2 On-line změna programu PLC

Mosaic umožňuje provádět on-line úpravy programu PLC bez zastavení řízení. Tato funkce vyžaduje podporu ze strany CPU a týká se pouze systémů TECOMAT TC700 a TC650. Kromě změn řídícího algoritmu lze přidávat a mazat proměnné, měnit jejich datový typ apod. Přepnutí mezi starým a novým programem je velmi rychlé, typicky méně než desetinu doby potřebné pro zpracování programu. Společně s možností vyměňovat I/O moduly PLC bez zastavení řízení je on-line změna programu důležitou podmínkou pro minimalizaci ztrát vzniklých odstavením řídicího systému při údržbě SW i HW PLC.

6.2.3 Komunikační možnosti:

Mosaic umožňuje komunikovat s řídicím systémem přes sériovou linku, Ethernet, USB. V prostředí je zahrnuta i podpora pro vytáčení připojení přes telefonní nebo GSM modem, a v poslední době oblíbené spojení přes Wi-Fi, jenž umožňuje dálkovou správu.[7]

7. GSM terminál Siemens TC35

GSM modem Siemens TC35 je zařízení pro bezdrátový přenos dat po sítích GSM mobilních telefonů. Je určen především pro přenos dat v průmyslu, automatizaci a řízení, kde je také využitelný jako univerzální část dalších zařízení. Vlastnosti zařízení jsou ve velké míře dány použitým GSM modulem Siemens TC35, který byl vybrán pro svou spolehlivost a dobré vysokofrekvenční parametry. Modem obsahuje vlastní čtečku SIM karty. Různé typy antén lze připojit anténním konektorem. Ovládání po RS-232 je obdobné jako u běžných modemů. Navíc, proti předchozímu typu, je implementováno automatické rozpoznávání komunikační rychlosti. Pomocí rozšířených AT příkazů lze zařízení využít jak pro přenos dat dle různých přenosových protokolů, tak pro zasílání a příjem SMS zpráv, faxu atd. Vestavěné hodiny a datum, stejně jako mód se sníženou spotřebou,dále rozšiřují možnosti využití tohoto modemu (aktivace modemu z režimu se sníženou spotřebou v zadaném čase, příchozím voláním nebo SMS zprávou, časová registrace volání atd.).

- **Základní technické parametry:**

Rozměry	104(123) x 70 x 53 mm
napájecí napětí	6 -24Vac nebo 8 - 30Vdc (nezáleží na polaritě)
odběr proudu	20 – 600 mA podle režimu provozu
SIM karta	3V/1,8V
datové rozhraní	RS-232
přenosová rychlost	max. 9600 b/sec
komunikační rychlost	nastavitelná 300b/s – 115kb/s, aut. rozpoznání komunikační rychlosti (od 4,8 kb/s)
vf přenosové pásmo	EGSM900. GSM1800
max. vf výkon	EGSM900 2W GSM1800 1W

- **Komunikační rychlost:**
  Jedná se o rychlost, kterou GSM modem komunikuje se zařízením, ke kterému je připojen pomocí RS-232. SM modem má autodetekci komunikační rychlosti (od 4,8 kb/s), pro použití nižší rychlosti je ji třeba naprogramovat podle komunikační rychlosti zařízení, ke kterému je modem přes RS-232 připojen.

- **Přenosová rychlost:**
  Rychlost, kterou GSM modem komunikuje se zařízením, ke kterému je připojen bezdrátově sítí GSM. Tato rychlost je pevně nastavena na 9600b.[11]

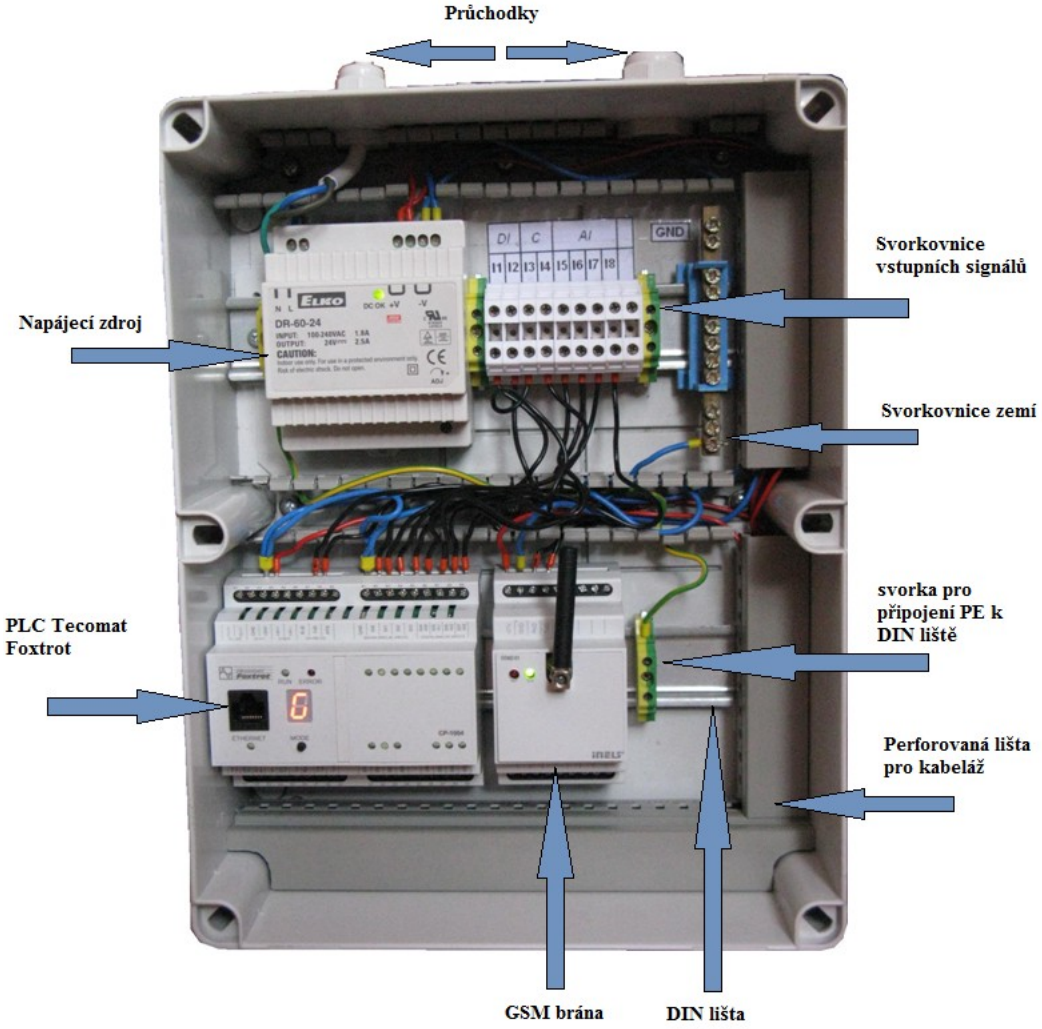
Obr. 7.1 Terminál siemens TC35 [11]

8. HW realizace měřicí stanice

Měřicí stanici netvoří pouze programovatelný automat, ale také napájecí zdroj, svorkovnice pro uživatelské připojení vstupních signálů a modul GSM Brány. GSM brána slouží k zasílání SMS zpráv na mobilní telefon, např. při vyžádání provedení zaslání aktuálních naměřených dat. Jelikož je vhodné, aby byla měřicí stanice kompaktní a přenosná, byly všechny komponenty nainstalovány do rozvaděčové krabice. Tato krabice rovněž zajišťuje velmi vysoké krytí celého zařízení, které má hodnotu IP55 (obr. 8.1.1).

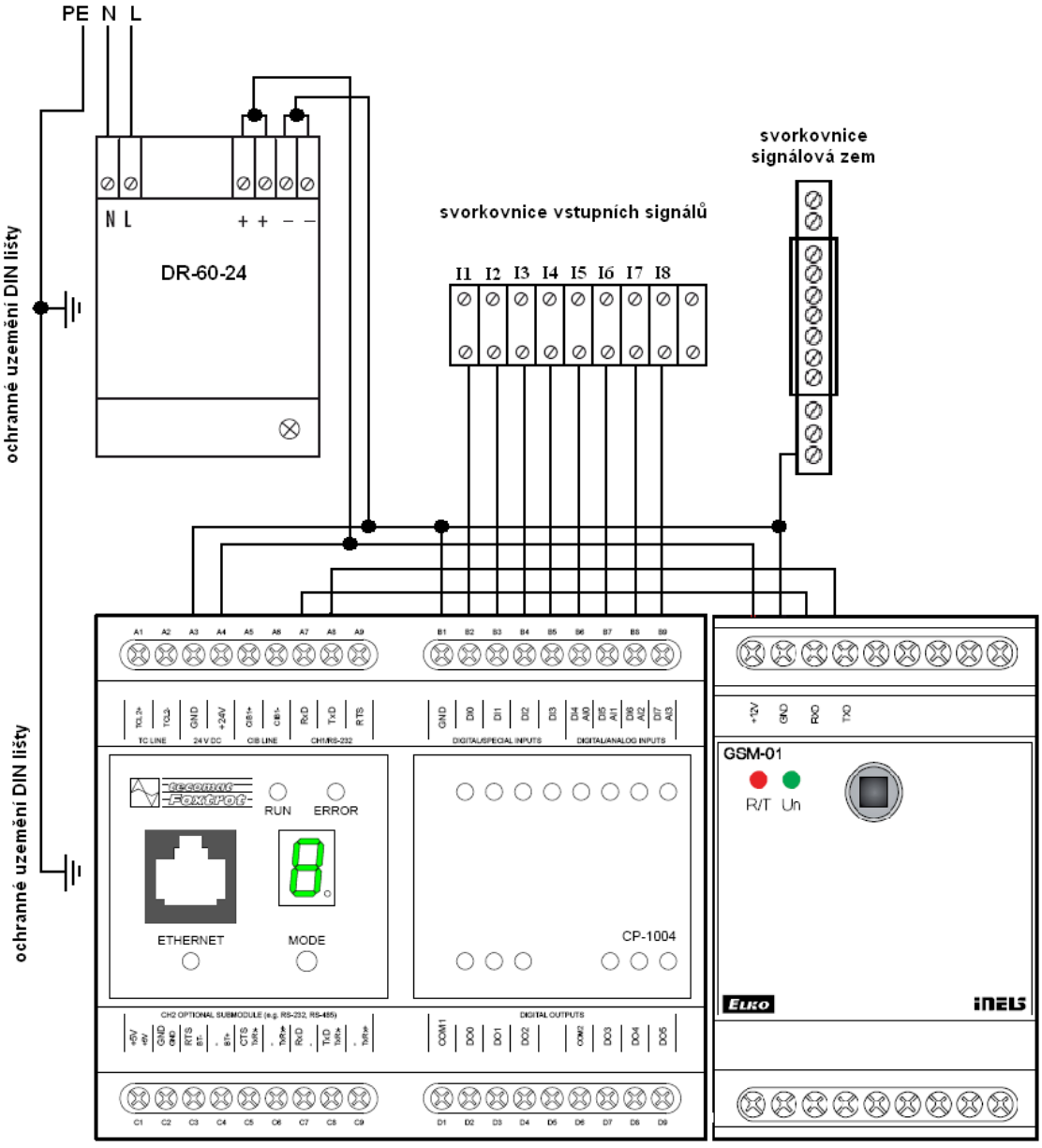
8.1 Popis zařízení

Pouzdro je tvořeno plastovou rozvaděčovou krabicí firmy ABB o rozměrech 320x247x120mm. Víko krabice je opatřeno těsněním, což zajišťuje vysoké krytí celého systému, a je vyrobeno z průhledného materiálu, takže je možno sledovat kontrolní LED diody všech přístrojů i za provozu. Jednotlivé prvky měřicí stanice jsou v krabici umístěny ve dvou řadách. Každá řada je tvořena DIN lištou, která slouží pro montáž přístrojů a dalších komponent.



Obr. 8.1.1 Stanice pro měření a archivaci dat TECOMAT FOXTROT.

V horní řadě vlevo je umístěn napájecí zdroj 24VDC firmy Elko. Napravo od zdroje je umístěna svorkovnice bílé barvy, která slouží pro uživatelské připojení jednotlivých měřených signálů. Ta je tvořena 9 piny, z nichž prvních 8 je zapojeno na 8 vstupních kanálů PLC. Devátý pin slouží pouze jako rezerva a není dosud zapojen. Jednotlivé signálové země všech signálů se pak připojují na můstkovou svorkovnici modré barvy, která je umístěna vpravo v horní řadě. V této horní řadě jsou dále použity tři kusy žlutozelené svorkovnice. První kus, umístěný úplně vlevo, zajišťuje připojení ochranného vodiče elektrorozvodné sítě k DIN liště. Další dva kusy slouží pouze k pevnějšímu uchycení jinak velmi vratké „signálové“ svorkovnice.



Obr. 8.2 Schéma elektrického zapojení měřicí stanice.[10]

V dolní řadě vlevo je opět na DIN liště umístěn základní modul PLC TECOMAT FOXTROT, konkrétně CP-1004 a napravo od něj GSM modul firmy Elko. Za tímto modulem je upevněn jeden kus

žlutozelené svorkovnice, která opět zajišťuje připojení ochranného vodiče elektrorozvodné sítě k této DIN liště.

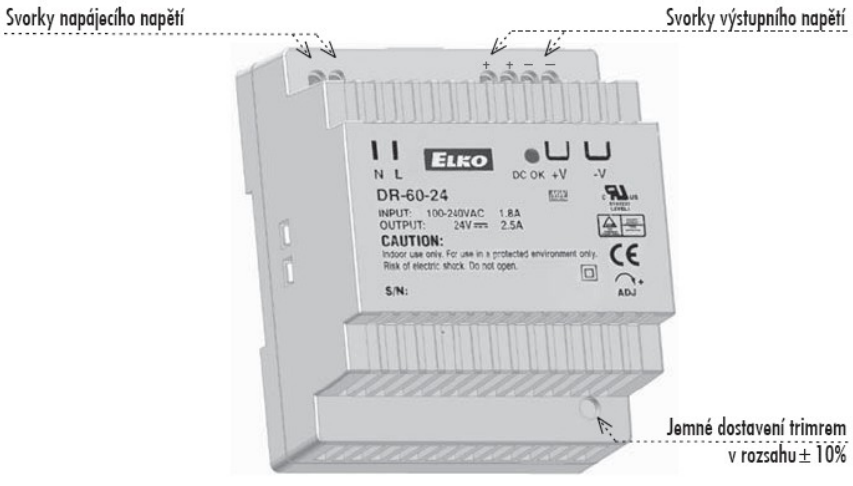
Veškerá kabeláž je uložena v perforované liště, která je vedena středem a po obvodu celé krabice, kromě její levé strany. Kabeláž je dovnitř krabice přivedena pomocí dvojice průchodek. Ty jsou umístěny v horní stěně krabice a jsou vyústěny přímo v perforované liště. Průchodka na levé straně slouží k přivedení napájecího kabelu. Průchodka na pravé straně pak k přivedení jednotlivých vodičů vstupních signálů. Obě průchodky jsou opatřeny těsněním a kónusovou maticí, která toto těsnění utáhne kolem kabelu. Průchodky tak nezhoršují krytí zařízení dané krytím krabice. Elektrické zapojení komponent je uvedeno na obr. 8.2 [10]

8.2 Napájecí zdroj

Napájecí zdroj DR-60-24 (obr. 8.2.1) je produktem firmy Elko. Jedná se o spínaný zdroj s výstupním napětím 24VDC a maximálním zatížením 2,5A. Některé technické parametry DR-60-24 jsou uvedeny v tab. 8.2 [6]

Napájecí napětí:	100-240 V AC/ 47-63 Hz
Výstupní napětí:	24 V
Max. zatížení:	2.5 A / 60 W
Zvlnění výstupního napětí:	150 mV
Účinnost:	84%
Tolerance výstupního napětí:	-0,01
Elektronická pojistka:	proti zkratu, přetížení a přepětí
Jemné dostavení výstupního napětí:	10% - trimrem
Přetížení:	do 105-160% jmenovitého výkonu
Krytí:	IP20

Tab. 8.2 Technické parametry DR-60-24.



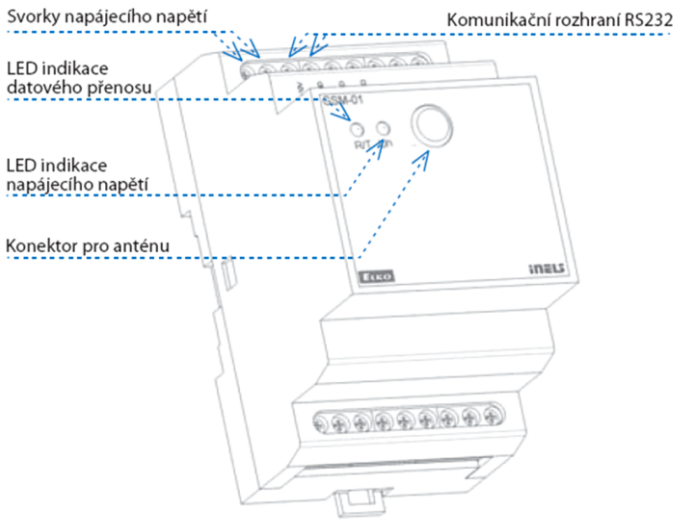
Obr. 8.2.1 Popis napájecího zdroje[6]

8.3 GSM brána

GSM brána GSM2-01 (obr. 8.3) je produktem firmy Elko a je součástí systému Inels, který slouží pro automatizaci budov. Tato GSM brána umožňuje přijímat SMS zprávy, jejichž pomocí je možno ovládat řídicí systém, nebo zasílat SMS zprávy informující uživatele o určitých událostech, které v systému nastaly. SIM karta se do jednotky vkládá po odejmutí čelního panelu. GSM2-01 se připojuje přímo k řídicí jednotce (v našem případě CP-1004) přes rozhraní RS-232. Některé technické parametry GSM2-01 jsou uvedeny v tab. 8.3.

Napájecí napětí/jm. proud:	24 V DC/400 mA
Indikace napájecího napětí:	zelená LED
Indikace výstupů:	červená LED
Spotřeba:	- Idle mode 25 mA
	- Speech mode 300 mA (průměr)
	- GPRS mode 360 mA (průměr)
	- Sleep mode 3,5 mA (max.)
	- Vypnuto 50 uA
Komunikační rozhraní:	1 x RS232, pro komunikaci s řídicím systémem
Počet informačních SMS:	max. 32
Počet přednastavených čísel:	max 16
Typ použitého GSM modulu:	Wireless Module MC39i
GSM síť:	Dual-band EGSM900 a GSM 1800
Instalace:	na DIN lištu EN 60715
Provedení:	3-MODUL
Stupeň krytí:	IP 20

Tab. 8.3 Technické parametry GSM2-01.



Obr. 8.3 Popis GSM brány GSM2-01.

8.3.1 Konfigurace GSM brány

Pro správnou funkci programu je třeba použít Knihovnu GSMLIB.mlb, která obsahuje podporu pro použití GSM brány GSM2-01. Ve verzi 1.5 umožňuje především příjem a odesílání krátkých textových zpráv SMS.

Pro odesílání a přijímání SMS slouží dva Funkční bloky

Funkční blok 1.2.1 SMS_Handler
SMS_Handler je funkční blok, zajišťující komunikaci s GSM bránou. Funkční blok pracuje nad strukturou komunikačního kanálu PLC, který musí být v režimu UNI s následujícími parametry:

- délka přijímací zóny 360 bytů,
- délka odesílací zóny 360 bytů,
- komunikační rychlost 9600 baudů,
- formát dat 8 bitů bez parity,
- maximální délka zprávy 360,
- minimální doba klidu na lince mezi přijímanými zprávami 5 bytů,
- minimální doba klidu na lince mezi odesílanými zprávami 40 bytů.

Pro správnou funkci je nutné, aby pro každý komunikační kanál byla pouze jedna instance funkčního bloku, která bude volána jednou během cyklu PLC.

Funkční blok SMS_Handler_2 je variantou bloku SMS_Handler využívající PDU módu komunikace s GSM bránou. SMS_Handler_2 rozšiřuje funkčnost bloku SMS_Handler o možnost volby kódování zpráv a vyžádání statusu o doručení zprávy. Kanál, se kterým blok pracuje, musí být v režimu UNI s následujícími parametry:

- délka přijímací zóny 380 bytů,
- délka odesílací zóny 360 bytů,
- komunikační rychlost 9600 baudů,
- formát dat 8 bitů bez parity,
- minimální doba klidu na lince mezi přijímanými zprávami 5 bytů,
- minimální doba klidu na lince mezi odesílanými zprávami 40 bytů.

Kódování SMS zpráv se řídí vstupy PlcCoding a SmsCoding.

Pro tuto diplomovou práci byl využit Funkční blok 1.2.1 SMS_Handler_2
Číslo střediska zpráv musí být uvedeno v mezinárodním formátu (tab 8.3.1).

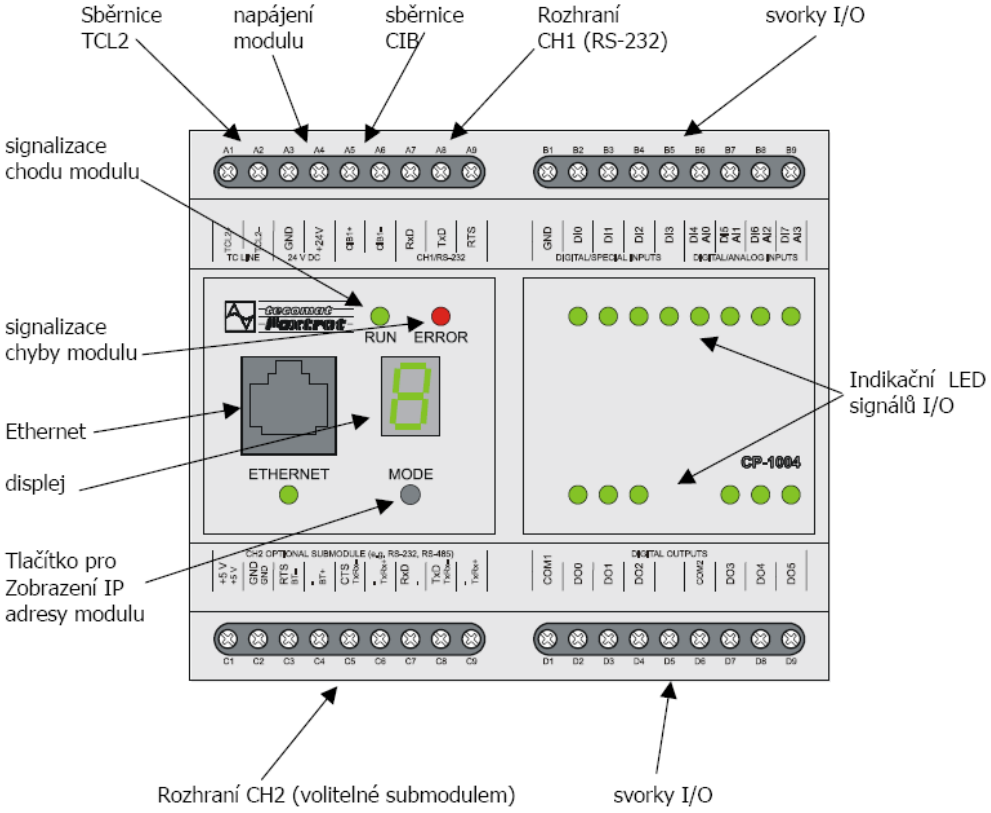
Operátor	Hodnota *SMSCenter*
O2	'+420602909909'
T-Mobile	'+420603052000'
Vodafone	'+420608005681'

Tab. 8.3.1. Čísla SMS center operátoru[9]

Odeslání SMS je prováděno přivedením signálu náběžné hrany na vstup Send. [9]

8.4 Centrální jednotka s CP1004

Periferní část modulů CP-10x4 tvoří deska IR-1057 (starší provedení IR-1055) obsahující 8 víceúčelových vstupů a 6 reléových výstupů. První čtyři vstupy DI0 - DI3 mohou být použity jako běžné binární vstupy nebo jako vstupy pro čítače. Další čtyři vstupy DI4 - DI7 mohou být použity jako běžné binární vstupy nebo jako analogové vstupy AI0 - AI3. Pod jménem IR-1057 (resp. IR-1055) se hlásí na systémové sběrnici procesor obsluhující tyto vstupy a výstupy. Pro jeho programování se používá vývojové prostředí Mosaic. Na čelním panelu je kromě vyvedeného rozhraní Ethernet k dispozici displej, který zobrazuje základní stav modulu a při držení tlačítka pod displejem ukáže aktuální IP adresu rozhraní Ethernet. Tento automat je navíc osazen submodulem pro rozhraní RS-232 na kanálu „CH2“ a flash pamětí 1GB (MMC karta).[4]



Obr. 8.4.1 Cetrální jednotka CP1004[4]

9. Aplikační software

9.1 Realizace aplikačního software

Pro vývoj aplikace bylo použito vývojového prostředí MOSAIC, verze 2.0.25.0 MOSAIC je vývojové prostředí pro všechny PLC TECOMAT a regulátory Tecoreg. Podporuje programování v jazycích podle normy IEC 61131-3 (IL, ST, LD, FBD) a firemní jazyk typu IL (mnemokód). [8]

9.2 Založení projektu a hardwarová konfigurace

Při založení projektu bylo vybráno programování podle normy IEC 61131-3 a programovací jazyk ST a FBD.

FBD - Function Block Diagram - jazyk funkčního blokového schématu. Vyjadřuje chování funkcí, funkčních bloků a programů jako soubor vzájemně provázaných grafických bloků, podobně jako v elektronických obvodových diagramech. Je to určitý systém prvků, které zpracovávají signály.

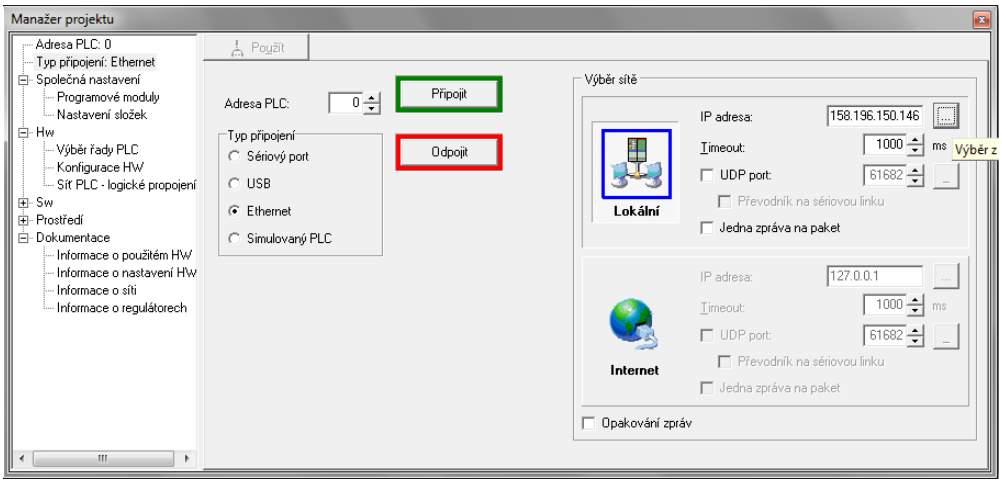
ST - Structured Text - jazyk strukturovaného textu. Velmi výkonný vyšší programovací jazyk, který má kořeny ve známých jazycích Ada, Pascal a C.

Samotné nastavení a HW konfiguraci PLC lze provést v prostředí MOSAIC v okně „Manažer projektu“.

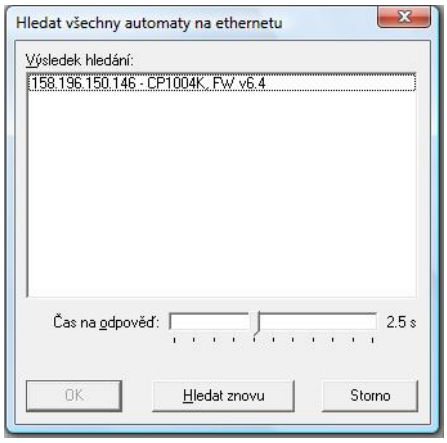
9.2.1 Manažer projektů

- **Typ připojení**

V záložce „Typ připojení“ je nutné nastavit typ připojení k PLC. Pokud je PLC připojeno pomocí komunikace „Ethernet“, je nutno zadat IP adresu PLC „158.196.150.146“. Poté se může k PLC připojit pomocí tlačítka „Připojit“. IP adresu PLC jde zjistit pomoci Tlačítka „Výběr z automatu připojených přes ethernet“, který je umístěn v „manažer projektu“ v záložce typ připojení“ vedle IP adresy (obr. 9.2.1.1). Pomocí tohoto tlačítka program Mosaic hledá připojené PLC přes komunikaci Ethernet, pokud je nalezena IP adresa, vypíše ji (obr. 9.2.1.2). Další možností je získat IP adresu přímo z PLC, kdy při držení stisknutého tlačítka „mode“ na PLC se postupně ukazuje IP adresa PLC.
Aby byla možná komunikace s PLC, musí být první trojčíslí IP adresy síťové karty v počítače a IP adresy PLC shodné.



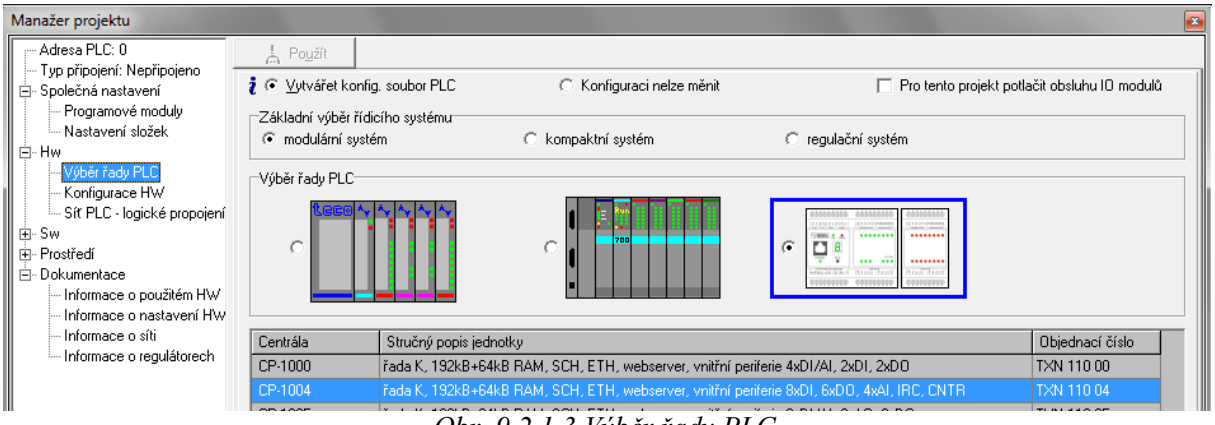
Obr. 9.2.1.1 Manažer projektu, typ připojení



Obr. 9.2.1.2 Hledáni IP adres

- **HW/Výběr řady PLC**

V záložce „HW/Výběr řady PLC“ je potřeba vybrat „modulární systém“, poté řadu PLC „FOXTROT“ a konkrétní typ „CP-1004“ (Obr. 9.2.1.3).

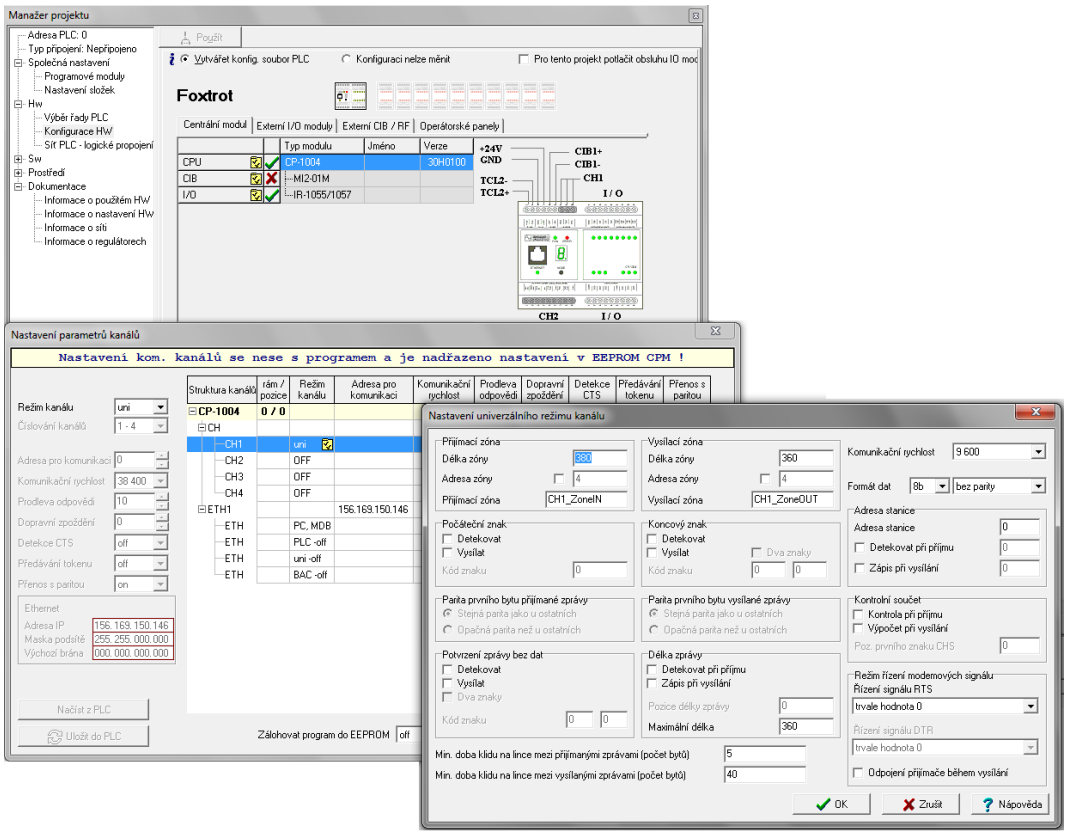


Obr. 9.2.1.3 Výběr řady PLC

- **HW/Konfigurace HW**

V záložce „HW/Konfigurace HW“ lze vyčíst nastavení přímo z PLC pomocí tlačítka „Načíst z PLC“, nebo nastavit ručně. Základní modul FOXTROT CP-1004 je zde složen ze tří částí: „CPU modul CP-1004, I/O modul IR-1055 a CIB modul MI2-01M“. Všechny části je možno povolit (zelená fajfka), zakázat (červený křížek) a konfigurovat (ikona žluté složky). Pomocí tlačítka „Příslušenství“ je možno vybrat sub modul. Použitý PLC je osazen sub modulem pro rozhraní RS-232, proto je zde pro „CH2“ vybráno „MR-0104 rozhraní RS-232 galvanicky oddělené“

Konfigurace „CPU“ (ikona žluté složky obr. 9.2.1.4) umožňuje nastavení různých komunikačních rozhraní. Konfiguraci je opět možno načíst z PLC pomocí tlačítka „Načíst z PLC“, nebo nastavení ručně editovat, a poté jej zapsat do PLC pomocí tlačítka „Uložit do PLC“. Komunikační kanál „CH1“ je potřeba nastavit do režimu „UNI“. Tento kanál je využit pro komunikaci s GSM bránou (obr. 9.2.1.4). Její nastavení je popsáno v kapitole 8.3.1.



Obr. 9.2.1.4 Nastaveni kanálů

Konfigurace „I/O“ (ikona žluté složky Obr. 8.2.1.4) umožňuje v jednotlivých záložkách nastavení binárních I/O, nastavení čítačů a nastavení analogových vstupů. Zde je důležité nastavení v záložce „Režim čítače“. „Režim čítače 2“ je nastaven na „dva jednosměrné čítače“. Tím se provede nastavení vstupů DI2 (na svorkovnici vstup „I3“) a DI3 (vstup „I4“) na čítačové vstupy s čítáním nahoru. V záložce „Analogové vstupy“ je provedeno povolení všech čtyř analogových vstupů a reprezentace dat z výstupu ADC je nastavena na: „Hodnota v inženýrských jednotkách (ENG)“. Tímto se provede povolení analogových vstupů AI0 až AI3 (na svorkovnici vstup „I5“ až „I8“) a předávaná hodnota analogových vstupů je v rozsahu 0 až 10,5, přičemž se jedná přímo o napětí [V].

Posledním důležitým nastavovacím prvkem je „Start PLC po zapnutí“, který je nutno nastavit na„Teplý“. Toto nastavení se provede v záložce „SW/Cpm“. Tímto zajistíme, že po vypnutí a znovu zapnutí PLC (např. při výpadku napájení) se zachová remanentní zóna paměti PLC a nedojde ke ztrátě dat.

9.3 Základní funkce aplikačního SW

Zadání diplomové práce požaduje získávání aktuálních naměřených hodnot pomocí GSM brány zasíláním SMS zpráv s hodnotami všech kanálů s datovou a časovou stopou. Další část se zabývá archivaci dat v režimu autonomním, tedy v režimu, kdy není měřicí stanice připojena k počítači. Data se budou ukládat z databoxu do paměťové karty po jeho naplnění, je to proto, abychom kartu nezničili stálým zapisováním hodnot v každém cyklu.

9.3.1Provozní režimy stanice

Provozní režimy měřicí stanice mohou být:

- **On-line provoz** - měřicí stanice je připojena k PC, po komunikační lince probíhá přenos aktuálních měřených dat do nadřízeného PC.

- **Autonomní provoz** - měřicí stanice (není připojena k PC) provádí měření a archivaci dat do paměti měřicí stanice, po připojení PC je možnost zpětného vyčtení archivů do databáze PC.

- **Konfigurace** - měřicí stanice je připojena k PC, ze servisní obrazovky PC je možná konfigurace a nastavení parametrů měřicí stanice.

V diplomové práci je využitá druha možnost - autonomní provoz. Data se ukládají na paměťovou kartu umístěnou v PLC, zpětně data vyčtou a zpracuji, nebo odešlou GSM bránou na terminál připojený k PC.

9.3.2 Typ vstupního kanálu

Z hlediska základního rozdělení kanálů, můžeme vstupní kanál nakonfigurovat jako:
- analogový
- digitální.

Analogový signál převádí spojitý signál na binární číslo s přesností danou rozlišením DAC. Digitální signál je pro určité rozmezí signálu vyhodnocen jako „log0“, pro jiné rozmezí jako „log1“. Mimo tato rozmezí nelze digitální signál určit, jedná se o zakázanou oblast.

Analogový kanál

Možné konfigurace pro signál analogového kanálu:

- napěťový
- proudový
- jiný

Jiným signálem je myšlena jakákoliv měřitelná veličina (teplota, radiace, vlhkost apod.), kterou lze převést na elektrický signál, proud nebo napětí. Stejně tak proudový signál lze převést na napěťový a naopak.

Možné konfigurace analogových vstupních kanálů podle úrovně signálu:

Napěťový:
- 0 až 10V
- -10 až 10V
- 0 až 5V
- rozsah pro Pt100
- rozsah pro Ni1000

Proudový:
- 0 až 20mA
- -20 až 20mA
- 4 až 20mA

Digitální kanál

Možné konfigurace pro signál digitálního kanálu:

- napěťový
- proudový

Možné konfigurace digitálních vstupních kanálů podle úrovně signálu:

Napěťový:
- 0 / 24 VDC
- 0 / 5 VDC
- 0 / 220 VAC
- 0 / 110 VAC

V případě proudového signálu se jedná o převod na napěťový signál.

Možné konfigurace digitálních vstupních kanálů podle jeho funkce:

- klasický DI
- DI pro zachycení krátkých impulsů, nebo přerušovací DI
- čítačový
- frekvenční

Čítačový kanál

Režimy čítačů:

- Jednosměrný čítač – využívá jeden vstupní kanál.
- Obousměrný čítač – využívá 2 vstupní kanály.
- Čítač s řízením směru – využívá 2 kanály.
- Obousměrný čítač s nulováním a zachycením – využívá 4 kanály.
- Čítač s řízením směru s nulováním a zachycením- využívá 4 kanály.

- Měření délky pulsů - využívá 4 kanály.
- Měření periody - využívá 4 kanály.
- Měření fázového posunu - využívá 4 kanály.

Nastavení kanálů

Nastavení kanálů jsou provedena v programu „LOGGER2.ST. Na obrázku 9.3.2.1 je zobrazeno nastavení kanálu K5, tedy prvního analogového kanálu AI1. Je potřeba nastavit číslo fyzického a logického kanálu a jeho rozsah. Další parametry k nastavení mohou byt jeho název, fyzikální jednotka, povolení a přiřazení simulované hodnoty.

1: prgCasovy.st | 2: prgMain.ST | 3: prgStudenyRestart.st | 4: prgTeplyRestart.st | 5: LOGGER2.ST | 6: CompareSMS.ST | 7: PrgSMS.FBD | WebMaker | 8: fbMakeSMS.ST |

```
// K5,AI datová struktura
fyzCisloK5       AT %R1600   : INT     :=0;    //číslo fyzického kanálu
logCisloK5       AT %R1602   : INT     :=5;    //číslo logického kanálu
nazevK5          AT %R1604   : STRING  :='analogový 1'; //název kanálu
jednotkaK5       AT %R1686   : STRING  :='[V]'; //fyzikální jednotka
aktivaceK5       AT %R1768.0 : BOOL    :=TRUE;   //aktivace měření kanálu
typDatK5         AT %R1769   : INT     :=3;    //výběr reprezentace dat ADC (hodnota ADC, procetna, v jednotkách vstup. signálu)
elRozsahHMK5     AT %R1771   : REAL    :=10.0;   //zadání horní meze rozsahu el. signálu (výstup z ADC)
elRozsahDMK5     AT %R1775   : REAL    :=0.0;   //zadání dolní meze rozsahu el. signálu (výstup z ADC)
techRozsahHMK5   AT %R1779   : REAL    :=10.0;   //zadání horní meze rozsahu tecnologického signálu (signál ve fyzikálních jednotkách)
techRozsahDMK5   AT %R1783   : REAL    :=0.0;   //zadání dolní meze rozsahu tecnologického signálu (signál ve fyzikálních jednotkách)
aktivaceSimulK5  AT %R1787.0 : BOOL;   //povolení ruční simulované hodnoty
hodnotaSimulK5   AT %R1788   : REAL;   //zadaní ruční simulované hodnoty
cisloArchivuK5   AT %R1792   : INT;    //přiřazení archivu k tomuto kanalu
```

Obr. 9.3.2.1 Nastaveni analogového kanálu

- Digitální kanály – DI1, DI2
  Úroveň signálu pro „log 0“ je 0V pro „log 1“ je 24V
  Perioda vzorkování:500ms

- Čítačové kanály – C1, C2
  Úroveň signálu pro „log 0“ je 0V pro „log 1“ je 24V
  Perioda vzorkování:500ms

- Analogové kanály – AI1, AI2, AI3, AI4
  Rozsah: 0-10V
  Perioda vzorkování:500ms

9.3.3 Autonomní režim

V autonomním režimu se řeší ukládání dat z PLC, které není připojeno k počítači. Data se ukládají do externí paměti PLC. Ukládaná data mohou být uložena v různých datových strukturách. V daném, případě se jedná o následující typ datové struktury:

- přesný čas a datum pořízení vzorku,
- vlastní naměřená hodnota.

Datová struktura ukládaných dat může vypadat takhle:

- rok
- měsíc
- den
- hodina
- minuta
- sekunda

- milisekunda
- vlastní naměřená hodnota

Datové typy jednotlivých parametrů jsou závislé na konkrétní konfiguraci typu kanálu.
Datové typy pro měřenou hodnotu mohou být:

- BOOLEAN pro samotný vstup DI
- INTEGER pro DI čítačový, frekvenční, nebo AI s informací vyjádřenou v rozsahu DAC

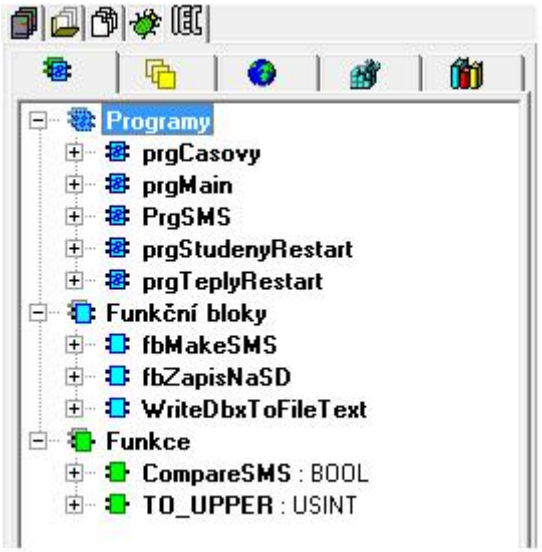
Hlavním požadavkem u autonomního režimu je, aby byla možnost pozdějšího vyčtení naměřených dat z paměti PLC do počítače.

9.3.4 On-line provoz

On-line provoz byl celý řešen prostřednictvím nadřazeného systému Promotic Ing. Jiřím Žáčkem v jeho diplomové práci s názvem „Návrh a realizace autonomní stanice pro měření a archivaci dat“. Pokud je spuštěn on-line režim, nadřazený systém si bude vyčítat data s datovou i časovou značkou a archivovat je na disk PC. Budou zobrazeny aktuální hodnoty měřicích kanálů.

9.3.5 Popis struktury aplikačního software

Aplikační software je vytvořen v projektové skupině „DiplomovaPrace“ a samotný projekt má název „Logger2“. Tento projekt se skládá z 5 souborů, 3 funkčních bloků a 2 funkcí zobrazených na obr. 9.3.5.1.



Obr. 9.3.5.1 Programové části softwaru

- **prg*.ST**

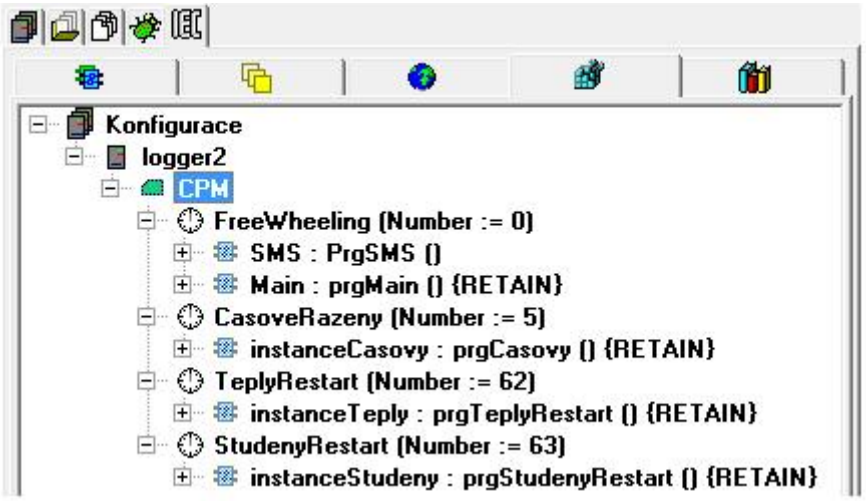
  Tyto soubory obsahují samotný kód programu, který je vykonáván v určitých procesech (úlohách). Každý z těchto souborů je tvořen programovou organizační jednotkou – PO, jedním programem, který tvoří stavební prvek uživatelského programu. Každý program, resp. instance programu, je přiřazen k určitému procesu, ve kterém se vykonává. TECOMAT FOXTROT umožňuje vytvořit až 65 procesů (P0 až P63), které jsou volány podle dané

strategie. Vytvořený aplikační program využívá procesy P0, P5, P62 a P63. Přiřazení jednotlivých programů k těmto procesům a způsob vykonávání těchto programů je uveden v tab. 9.3.5

Proces	Přiřazený program	Způsob vykonávání procesu (programu)
P0	prgMain	základní proces volaný v každém cyklu
P0	PrgSMS	základní proces volaný v každém cyklu
P5	prgCasovy	časově řazený proces (každých 400ms)
P62	prgTeplyRestart	proces zařazený po teplém restartu
P63	prgStudenyRestart	proces zařazený po studeném restartu

Tab. 9.3.5 Procesy využité v aplikačním software

Vytvoření jednotlivých procesů instancí programů a jejich přiřazení k programům v prostředí MOSAIC je zřejmé z obr. 9.3.5.2. Nejprve je potřeba vytvořit program, poté je možno pomocí vyplnění konfiguračních dialogů aktivovat příslušný proces, vytvořit instanci programu, která se bude v tomto procesu vykonávat, této instanci přiřadit vytvořený program (POU). Systém automaticky generuje toto nastavení. Tento kód pak představuje předpis celého programu pro PLC.[11][8]



Obr. 9.3.5.2 Konfigurace programu.

- **prgMain.st**

  Tento soubor obsahuje hlavní část programu. Je zde prováděno načtení vstupních hodnot jednotlivých kanálů, nastavováni hodnot podle zadaných konfiguračních parametrů z nadřazeného systému a samotná archivace těchto dat do paměti PLC.

- **prgCasovy.st**

  Soubor obsahuje části kódu, které není zapotřebí provádět v každém cyklu programu. Zde se provádí restartování dataloggeru, nastavení do výchozího stavu a ukončení měření.

- **prgTeplyRestart.st a prgStudenyRestart.st**

  Soubory obsahující kódy k ošetření některých stavů, které nastanou po spuštění PLC.
- **LOGGER2.ST**

  Tento soubor obsahuje deklarace globálních proměnných programu a definici uživatelských datových typů. Veškeré globální proměnné jsou deklarovány v remanentní zóně paměti.
- **prgSms.FBD**

  Soubor obsahuje nastavení GSM brány a samotné provedení posílání SMS zpráv, viz kapitola 8.4. Je vytvořen v jazyce FBD, další jeho částí jsou vnořené funkce pro porovnávání příchozí SMS „compareSMS.ST“ a „fbMakeSMS“, kde se nastavuje formát odesílané SMS. V hlavním programu „prgMain.st“ je potřeba nastavit důležité parametry pro zasílání zpráv-viz kapitola 9.4.
- **fbMakeSMS.ST**

  V tomto funkčním bloku je naprogramován formát zasílané SMS s hodnotami vstupních kanálů, doplněn datem a časovou stopou.
- **fbZapisNaSD.FBD**

  V tomto funkčním bloku se vytváří soubor na paměťové kartě, do kterého budeme ukládat data z databoxu. Vytváří se zde název souboru a přiřazuje se hodnota zapisovaná do souboru.
- **WriteDbxToFileText.ST**

  Část programu, ve které je vytvořen přepis obsahu databoxu na paměťovou kartu po zaslání řídicí SMS. Je zde vytvořena struktura ukládaných dat viz kapitola 9.5.
- **CompareSMS.ST**

  V této funkci je porovnáván text příchozí SMS s textem řídících SMS. Pokud se jedná o řídicí SMS, provede se její funkce.
- **TO_UPPER.ST**

  Pomocí této funkce se převádí příchozí textový řetězec na velká písmena.

9.4 GSM Brána aplikační software

Program pro ovládání GSM brány byl vytvořen v jazyce FBD (Function Block Diagram), který vyjadřuje chování funkcí, funkčních bloků a programů jako soubor vzájemně provázaných grafických bloků, podobně jako v elektronických obvodových diagramech.

Při komunikaci mezi GSM bránou a PLC byla objevena chyba v sériové komunikaci modulu GSM brány. Konzultací s firmou Teco, a.s. byla chyba identifikována a zajištěna oprava stávajícího modulu. Po dobu opravy tohoto modulu, mi firma Teco, a.s. zapůjčila nový modul GSM brány.

Program je vytvořen tak, že pokud uživatel pošle SMS na SIM kartu v GSM bráně (která je umístěna v měřicí stanici) s textem "POSLI", obratem se pošle SMS s aktuálními hodnotami měřicí stanice opatřené časovou stopou na SIM kartu umístěnou v GSM terminálu nebo do mobilního zařízení. Vyčítání je prováděno ve vizualizační části popsané v kapitole 10. K aktivaci vstupních kanálů, deaktivaci měřicí stanice a ukládání dat slouží další řídicí SMS.

Seznam řídicích SMS:

- „Poslat" -text SMS slouží k zaslání SMS s aktuální hodnotou na všech vstupních kanálech
- „StartAI" -text SMS pro aktivaci analogových kanálů
- „StartC" -text SMS pro aktivaci čítačových kanálů
- „StartDI" -text SMS pro aktivaci digitálních kanálů
- „StopAI" -text SMS pro deaktivaci analogových kanálů
- „StopC" -text SMS pro deaktivaci čítačových kanálů
- „StopDI" -text SMS pro deaktivaci digitálních kanálů
- „Uloz AI" -text SMS pro uložení hodnot analogových vstupů z databoxu na paměťovou kartu
- „Uloz C" -text SMS pro uložení hodnot čítačových vstupů z databoxu na paměťovou kartu
- „Uloz DI" -text SMS pro uložení hodnot digitálních vstupů z databoxu na paměťovou kartu

U řídicích SMS nezáleží na velikosti písmen, všechny příchozí textové řetězce jsou pomocí funkce „lng := TO_Upper" převedeny na velká písmena. Řídicí SMS jsou vytvořeny ve funkci compareSMS.

Dalším způsobem zasílání SMS zprávy s aktuálními hodnotami je možné řešit automatickým odesíláním v předem daných časových intervalech.

9.4.1. Popis programu pro SMS bránu

- **FB SMS_HANDLER_2**

Hlavním funkčním blokem je použit SMS_HANDLER_2, jehož nastavení je popsáno v kapitole 8.3.1. Funkční blok slouží k posílání SMS přivedením náběžné hrany signálu na vstup „SEND". U funkčního bloku je potřeba nastavit parametry důležité pro přenos dat. Hlavním parametrem je číslo SIM karty, na kterou chceme posílat SMS v proměnné „g_Recipient", PIN kód v proměnné „g_Pin" a číslo střediska zpráv SIM karty umístěné v GSM bráně. Všechny tyto proměnné je potřeba nastavit v souboru „LOGGER2.ST" ve spodní části programu.

```
VAR_GLOBAL
  SendStatus : BOOL;
  g_RecvSMS : SMS_STRING;
  g_Pin : PIN_STRING :='xxxx';
  g_SMSCenter : NUMBER_STRING :='+420608005681';
  g_Sender : NUMBER_STRING;
  g_Recipient : NUMBER_STRING :='+420608115682';
  g_SMSText : SMS_STRING;
  g_Caller : NUMBER_STRING;

END_VAR
```

Obr. 9.4.1 Nastaveni proměnných pro odesílání dat

- **FBmakeSMS**

Před funkčním blokem SMS_HANDLER_2 je vytvořena funkce „FBmakeSMS“, po jejím otevření se dostaneme do programu „FBmakeSMS.ST“, kde je naprogramovaný text posílané SMS, přiřazení hodnot a jejich datové typy.
Do proměnné Comma se nastavuje oddělující znak mezi hodnotami naměřených veličin, v práci byl použit znak „ ; “. Nejprve se pomocí funkce GetDateTime vytvoří formát aktuálního data ve formátu „rok, měsíc, den, hodina, minuta, sekunda“, poté se oddělí „ ; “ a vypisuje se první hodnota z digitálního vstupu DI1. Tato hodnota nabývá pouze dvou hodnot „0“ a „1“, převádí se z formátu BOOL do formátu STRING a zabírá jen jedno místo v řetězci textu SMS. Následuje hodnota DI2, která je také ve stejném formátu. Další dvě proměnné jsou hodnoty z čítače C1 a C2, které se převádí z formátu DINT do STRING o délce řetězce až 20 znaků. Posledními čtyřmi proměnnými jsou hodnoty z analogových vstupů AI1-AI4, zapisované z typu REAL do formátu STRINGF, jenž je naformátován na pět míst s jedním desetinným místem odděleným “.“. Kdybychom do uvozovek napsali '%5.3f', bude hodnota vypadat takto 0.999, jedním znakem je započítána desetinná tečka. Celková velikost SMS je maximálně 128 znaků.

```
SMSText := DT_TO_STRINGF(in := GetDateTime(), format := '%TYYYY-MM-DD-hh:mm:ss') +
           Comma + BOOL_TO_STRING(realArchivDI.hodnotaK1) +
           Comma + BOOL_TO_STRING(realArchivDI.hodnotaK2) +
           Comma + DINT_TO_STRING(realArchivC.hodnotaK3) +
           Comma + DINT_TO_STRING(realArchivC.hodnotaK4) +
           Comma + REAL_TO_STRINGF(in := realArchivAI.hodnotaK5, format := '%5.1f') +
           Comma + REAL_TO_STRINGF(in := realArchivAI.hodnotaK6, format := '%5.1f') +
           Comma + REAL_TO_STRINGF(in := realArchivAI.hodnotaK7, format := '%5.1f') +
           Comma + REAL_TO_STRINGF(in := realArchivAI.hodnotaK8, format := '%5.1f');
```

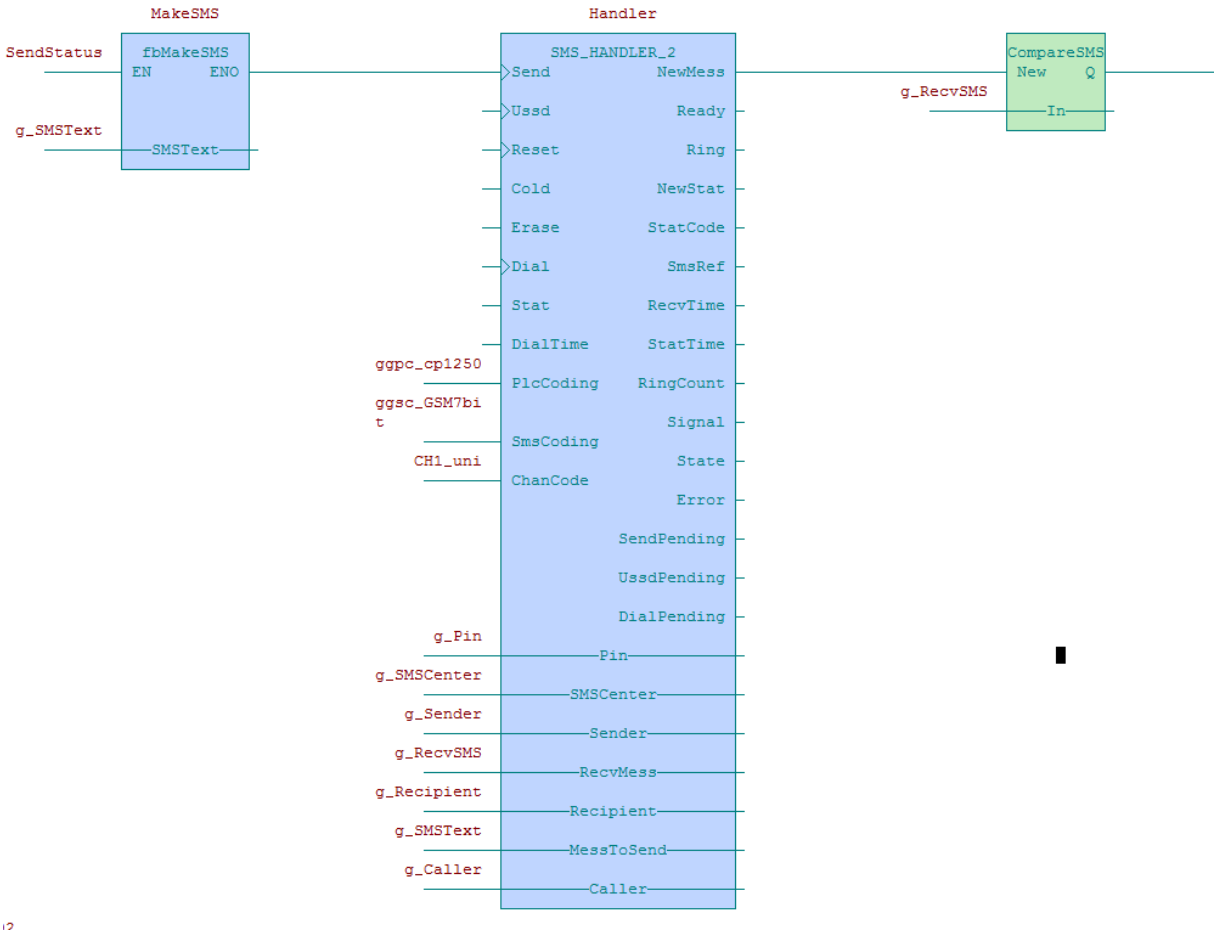
Obr. 9.4.2 Nastavení textu odesílané SMS

Text přijaté SMS je ve formátu:
2011-04-25-17:30:56;1;0;0;0; 5.2; 0.0; 0.0; 0.0,
kde je zobrazeno nejprve datum, čas a hodnoty DI1, DI2, C1, C2, AI1, AI2. AI3, AI4.

- **compareSMS**

Funkce compareSMS porovnává, zda byl text přijaté SMS shodný s textem uloženým v programu „compareSMS.ST“ Pokud se jedná o text „POSLAT“, vyšle signál „SEND“ a odešle se SMS. Pokud se jedná o jiný řídicí signál, provede danou operaci, např. aktivace analogových vstupů. Ukázka hlavní části programu, je na obr 9.4.2. Funkce obsahuje také i ostatní řídicí kódy, které zajišťují zahájení měření jednotlivých kanálů, jejich deaktivaci a funkce pro archivaci dat z databoxu přepsáním na paměťovou kartu. Řídicí signály jsou popsány.

V dalších funkcích programu se řeší vynulování povelu „SendStatus“, tak, aby nedošlo k opětovnému posílání dat v dalším cyklu, a vymaže se přijatá SMS.



Obr. 9.4.2 Hlavní část funkčního bloku SMS brány

9.5. Práce s paměťovou kartou

Ve stanici pro měření a archivaci dat TECOMAT FOXTROT je umístěna 1GB paměťová karta, do ní se zapisují archivovaná data z měřicí stanice. Data se dají přenést do počítače pomoci ethernetu.

Program pro archivaci dat do paměťové karty umístěné v PLC byl vytvořen v dvou programovacích jazycích. Část, ve které je vytvářená struktura ukládaných dat a zapisování hodnot do databoxu, je vytvořena v jazyce ST, jedná se o funkční blok „WriteDbxTiFileText.ST“. Druhá část programu je vytvořena v jazyce FBD ve funkčním bloku „fbZapisNaSD.FBD“. Zde je provedeno samotné nastavení adresáře a název souboru, kam se mají data ukládat při přepisování z databoxu na paměťovou kartu.

Po zapnutí měřicí stanice a aktivace příslušných kanálů řídicími SMS („StartAI“, „StartDI“, „StartC“) systém začne snímat vstupní signály podle zvolené periody vzorkování. Měřená data jsou ukládána do databoxu v PLC. Velikost databoxu je pro digitální a čítačové vstupy o velikosti 150 000b a délka jednoho zápisu těchto kanálu je 15b. Velikost databoxu pro analogové vstupní signály je 212 000b a délka zápisu je 23b.

Archivace dat probíhá dvěma způsoby:

- Při naplnění databoxu

  V tomto případě dochází k přesunu dat z databoxu do souboru na paměťovou kartu, v době, kdy je databox naplněn. Velikost databoxu a doba potřebná pro jeho naplnění je uvedena v tabulce 9.5.1.

- Na požádání

  Zde se provádí přenos dat z databoxu na paměťovou kartu pomocí řídicích SMS. Při zaslané SMS ve tvarech „Uloz DI“, „Uloz C“ a „Uloz AI“ se provede přepis databoxu na paměťovou kartu. Tohoto způsobu se využívá při měřeních s velkou periodou vzorkování, kdy dochází k naplnění databoxu v několika hodinách nebo dnech – viz tabulka 9.5.1. Jedná se například o měření emisí škodlivých látek ve vzduchu, které se provádí 1x denně.

Perioda vzorkování [s]	Archív digitálních dat Doba potřebná k naplnění [hod]	Archív digitálních dat Doba potřebná k naplnění [hod]	Archív digitálních dat Doba potřebná k naplnění [hod]
0,1	0,27	0,27	0,25
0,5	1,38	1,38	1,28
1	2,8	2,8	2,56
10	28	28	25,6
60	166,7	166,7	153,6
600	1667	1667	1536

Tab. 9.5.1 tabulka s dobou potřebnou k naplnění databoxu jednotlivých vstupů

9.4.1. Popis programu pro ukládání dat na paměťovou kartu

- **„WriteDbxTiFileText.ST“**

  Zde je uložen kód programu, který vytváří strukturu ukládaných dat a zapisuje data do databoxu. Data jsou poté připravena pro zápis na paměťovou kartu.

  Struktura ukládaných dat je ve formátu:

  - Rok
  - Měsíc
  - Den
  - Hodina
  - Minuta
  - Sekunda
  - Milisekunda
  - Data1
  - Data2

U analogových dat je struktura o 2 datové zápisy delší, protože analogové kanály mají 4 vstupy. Na obrázku 9.4.1.1 je zobrazen část kódu programu pro přepis dat a vytvoření struktury zapisovaného řetězce do databoxu.

```
IF NOT iWriteToFile.done AND NOT iWriteToFile.busy THEN //pokud neukladam pripravit zaznam
  n  := (size-savedsize) / sizeofstruct;
  na := min(n1, n);
  savedsize := savedsize + USINT_TO_UDINT(ReadBlockFromDBx(variable := void(bufferbin), dataBoxAddress := srcAdr + savedsize, length :=
  IF na > 0 THEN
    j := 0;
    FOR i := 0 TO UDINT_TO_UINT(na-1) DO
      ptrDI := ADR(bufferbin) + UINT_TO_UDINT(i)*sizeofstruct;
      ptrC  := ADR(bufferbin) + UINT_TO_UDINT(i)*sizeofstruct;
      ptrAI := ADR(bufferbin) + UINT_TO_UDINT(i)*sizeofstruct;

      sline :=   USINT_TO_STRINGF(in := ptrDI^.rok, format := '20%02d-')+ USINT_TO_STRINGF(in := ptrDI^.mesic, format := '%02d-') +
                 USINT_TO_STRINGF(in := ptrDI^.den, format := '%02d-') + USINT_TO_STRINGF(in := ptrDI^.hodina, format := '%02d:') +
                 USINT_TO_STRINGF(in := ptrDI^.minuta, format := '%02d:') + USINT_TO_STRINGF(in := ptrDI^.sekunda, format := '%02d:') +
                 USINT_TO_STRINGF(in := ptrDI^.milisek, format := '%02d;');

      CASE datatype OF
        0 : sline := sline + DINT_TO_STRING(ptrDI^.hodnotaK1) + ';' + DINT_TO_STRING(ptrDI^.hodnotaK2) + '$r$n';
        1 : sline := sline + DINT_TO_STRING(ptrC^.hodnotaK3) + ';' + DINT_TO_STRING(ptrC^.hodnotaK4) + '$r$n';
        2 : sline := sline + REAL_TO_STRING(ptrAI^.hodnotaK5) + ';' + REAL_TO_STRING(ptrAI^.hodnotaK6) + ',' +
                             REAL_TO_STRING(ptrAI^.hodnotaK7) + ';' + REAL_TO_STRING(ptrAI^.hodnotaK8) + '$r$n';
      END_CASE;
```

Obr. 9.4.1.1 Část kódu vytvoření struktury a zápis dat do databoxu.

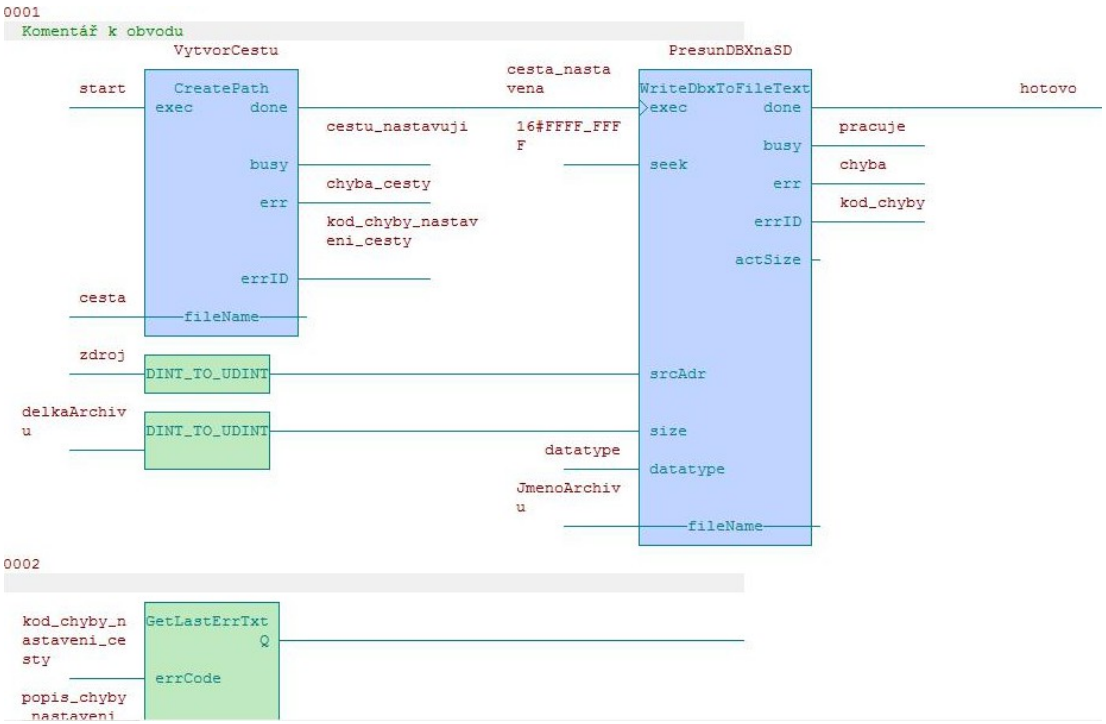
- **„fbZapisNaSD.FBD“**

  V tomto funkčním bloku je provedeno nastavení cesty pro zapisování souboru a jeho název. Nastavuje zápis dat do paměťové karty a kontroluje, zda byl přenos proveden.

  Ve funkčním prvku „VytvorCestu“ se vytváří cesta do složky na paměťové kartě, kde budou data ukládána. Kontroluje se zde, pokud je cesta a soubor vytvořen, pokud není, vytvoří jej.

  Dalším prvkem je „přesunDBXnaSD“, zde je potřeba nastavit, kam se mají data uložit, zdroj dat, typ a jméno archívu. Důležitým parametrem je parametr „seek“, ve kterém se nastavuje, jakým způsobem se budou data do souboru ukládat. Mohou být 3 možnosti jak data ukládat, na začátek souboru, doprostřed, nebo na konec souboru. Hodnota 16#FFFF_FFF nastavuje typ ukládání za poslední řádek v souboru. Do proměnné „cesta“ je v hlavni části programu přiřazena hodnota WWW/DATA, v této složce na paměťové kartě jsou data uložena. Proměnná „jmenoArchivu“ může nabývat 3 hodnot: „ARCHIVAI“, ARCHIVDI“, „ARCHIVC“. Tyto hodnoty jsou přiřazovány podle typu dat ukládaných na paměťovou kartu a nastavují jméno souboru s daty.

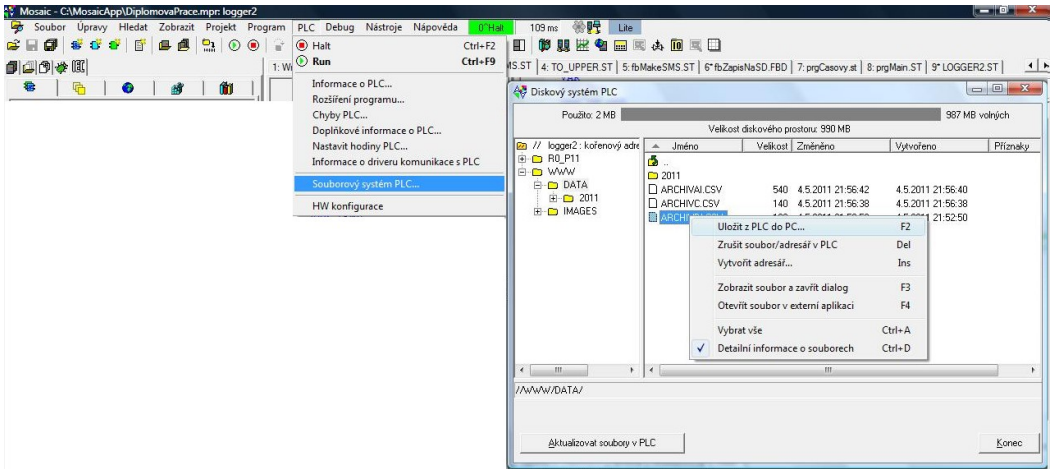
  Další prvky „GetLastErrTXT“ kontrolují, zda je správně nastavena cesta k souboru a správný zápis souboru. Pokud není žádná chyba, data jsou přepsána do souboru na paměťovou kartu a je vypnuto kopírování dat z databoxu. Na obrázku 9.4.1.2 je uvedena hlavní část funkčního bloku.



Obr. 9.4.1.2 Hlavní část funkčního bloku fbZapisNaSD.FBD

9.4.2. Vyčítání dat z paměťové karty

K vyčítání dat z paměťové karty musí být PLC připojeno k počítači. Komunikace mezi PLC a PC je nastavena v režimu „HALT“, tedy vypnutí běhu programu v PLC. V hlavní nabídce je potřeba vybrat záložku „PLC“, poté vybrat „Souborový systém PLC…“. Otevře se okno „diskový systém PLC“, které nám zobrazuje data uložená na paměťové kartě. Pravým kliknutím myši na vybraný soubor se zobrazí nabídka činnosti, ze které si vybereme uložení dat do PC. Zadáme, kam chceme soubor uložit, a zmáčkneme tlačítko „ULOŽIT“. Celý postup je uveden na obrázku 9.4.2.1



Obr. 9.4.2.1 Postup ukládání dat z PLC do PC

10. Vizualizační software

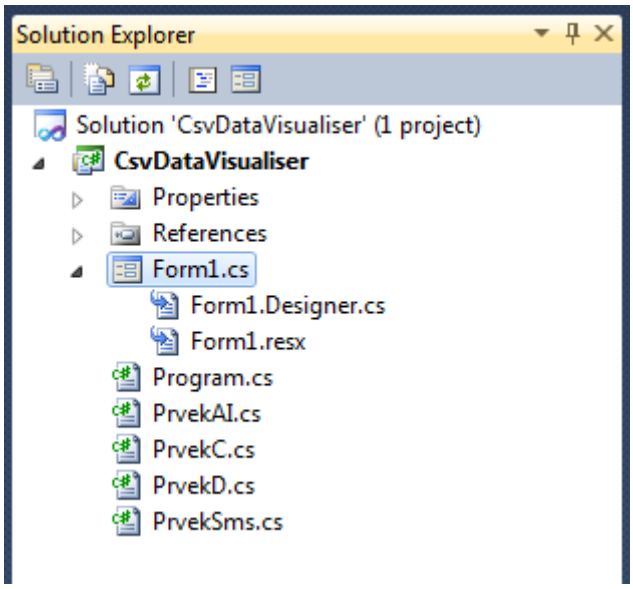
Vizualizační část programu slouží k načítání dat ve formátu CVS uložených buď v paměti PLC, které je nutno ručně zkopírovat do počítače, nebo dat přijatých pomoci GSM terminálu, k jejich zobrazení a grafickému zpracování v počítači.

Data ve formátu CSV jsou uložena v textovém řetězci a každá hodnota je oddělena „ ; “.

Vizualizační část je řešena v programu C#, ve vývojovém prostředí Microsoft Visual C# 2010 expres. Program načítá data vybraná uživatelem z počítače, vypíše je na obrazovce v tabulce dat a poté zobrazí graf daných hodnot, který lze uložit jako obrázek ve formátu „jpg“.

10.1 Struktura aplikačního softwaru

Aplikační software je vytvořen v projektové skupině „CSVDataVizualzer“. Tento projekt se skládá ze šesti souborů zobrazených na obrázku 10.1.1.

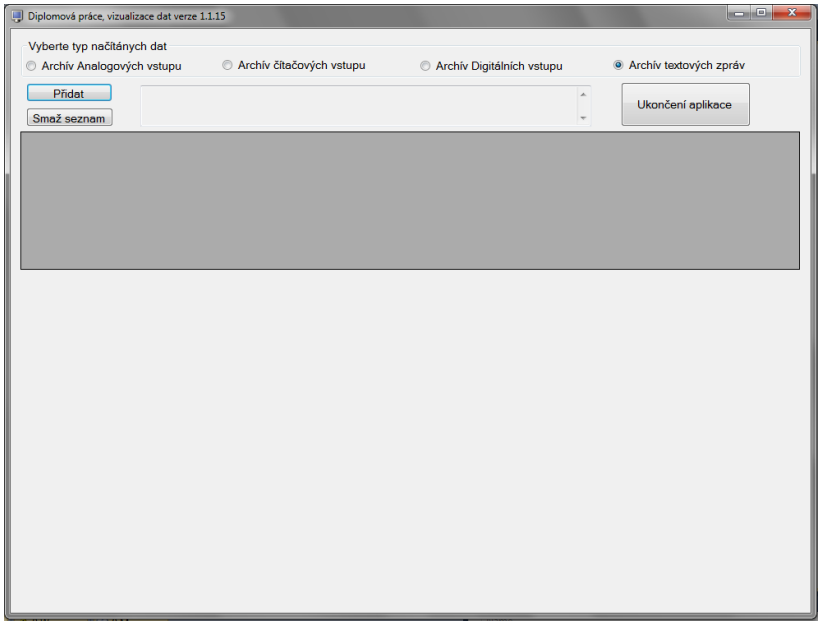


Obr. 10.1.1 Struktura aplikace

- **Form1.cs**
  Hlavní část programu je vytvořena v „Form1.cs“, ve kterém probíhá načítání dat, vypisování dat do tabulky ve vnořeném programu Microsoft Office Excel a zobrazování pomoci grafu.

- **Form1.Designer.cs**
  Vizualizační část programu je vytvořena v souboru „Form1.Designer.cs“, který obsahuje (obr. 10.1.2):

  - výběr načtení typu dat pomocí „radionbutton“,
  - dvě tlačítka „Přidat“ a „Smaž seznam“,
  - textový panel zobrazující nahrané soubory s jejich cestou umístění v počítači,
  - tabulková část zobrazených dat,
  - grafické zpracování.



Obr. 10.1.2 Vizualizace dat

10.2 Popis funkcí vizualizační části aplikačního softwaru

- **Výběr z typu dat**

  Pomocí „radionbutton“ Archiv analogových vstupů“, „Archiv čítačových vstupů“, „Archiv digitálních vstupů“ a „Archiv textových zpráv“, si vybíráme typ načtených dat, která se následně zobrazí v tabulce a grafickém zpracování.

  Pokud se vybere jiný typ dat, nebudou načtena a zobrazí se hlášení o chybě. Datová struktura všech kanálů je odlišná: mají jinou délku hodnot, jiný je počet kanálů, a tím i počet sloupců v tabulce.

  Hodnoty analogových vstupů mají datovou strukturu: 2011-04-21-14:05:30;5;8;4;1
  Hodnoty digitálních vstupů mají datovou strukturu: 2011-04-21-14:07:32;1;0
  Hodnoty čítačových vstupů mají datovou strukturu: 2011-04-21-14:05:32;20;40
  Hodnoty SMS mají datovou strukturu: 2011-04-25-17:30:56;0;0;0;0; 5.2; 0.0; 0.0; 0.0

  Prvním parametrem je datum a čas, ten je u všech stejný. Poté se struktura liší, u AI jsou 4 hodnoty analogových vstupů, u D a C jen 2 hodnoty vstupů a u SMS jsou postupně zapsány všechny kanály měřicí stanice.

- **Tlačítko „Přidat“**

  Při stisknutí tlačítka se pomocí funkce „Openfiledialog“ otevře okno, ve kterém zadáme cestu k souboru s hodnotami, které chceme načíst. Soubory k načtení musí být ve formátu *.CSV, nebo ve formátu *.TXT, protože formát CSV je textový řetězec oddělen znaménkem „;“. Textový řetězec může byt uložen také ve formátu *.TXT.

Data jsou na paměťovou kartu ukládána ve více souborech, vždy po naplnění databoxu je potřeba načítat více souborů. Proto je program navržen tak, abychom mohli po prvním zadání cesty k souboru zadat druhou cestu k souboru s hodnotami atd. Pokud zadáme špatnou cestu k souboru, nebo špatný typ dat, program vypíše chybnou hlášku ve vyskakovacím okně.

- **Tlačítko „Smaž seznam“**

  Tímto tlačítkem smažeme všechny soubory, které jsou načtené. Je to nutné proto, abychom mohli přejít na jiný typ dat, která chceme načíst. Data z kanálu digitálního mají jiný formát textu než data získaná z kanálu analogového. Toto tlačítko je nutno zmáčknout před každým měněním typu načítání dat.

- **Textový panel**

  V tomto textovém panelu se zobrazují načtená data s jejich adresovou cestou k umístění v počítači. Lze zjistit, které soubory jsou načteny a které ne.

- **Tabulková část zobrazených dat**

  V této části se zobrazuje tabulka načtených dat. Data jsou zobrazována pomocí vnořeného programu Microsoft Office Excel, podle vybraného zobrazení vstupních kanálů. Obrázek 10.2.1 zobrazuje část kódu v programu C# pro vytvoření řádku v tabulce podle hodnot analogových vstupů. Tabulka je řazena podle prvního sloupce načtených hodnot, tedy data a času, proto nerozhoduje, zda načtené soubory s hodnotami řadíme podle data pořízení. Další sloupce jsou pojmenovány podle názvu hodnot.

```
private void VlozHodnotyAIDoExcelu(ref Excel.Worksheet xlWorkSheet)
{
  xlWorkSheet.Cells[1, 1] = "Datum";

  for (int i = 2; i < 6; i++ )
    xlWorkSheet.Cells[1, i] = String.Concat("AI", i - 1);

  int radek = 2;
  foreach (PrvekAI prvekAI in dataAI.OrderBy(prvek => prvek.Date.Ticks))
  {
    xlWorkSheet.Cells[radek, 1] = prvekAI.Date.ToString("yyyy-MM-dd-HH:mm:ss");
    xlWorkSheet.Cells[radek, 2] = prvekAI.AI1;
    xlWorkSheet.Cells[radek, 3] = prvekAI.AI2;
    xlWorkSheet.Cells[radek, 4] = prvekAI.AI3;
    xlWorkSheet.Cells[radek, 5] = prvekAI.AI4;

    radek++;
  }
}
```

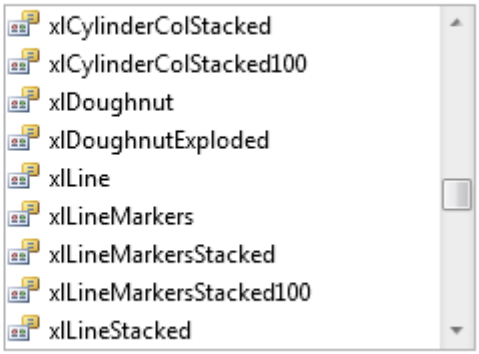
Obr. 10.2.1 Ukázka kódu zápisu analogových dat do tabulky

- **Grafické zpracování**

  V této části aktivního okna je zobrazen graf z načtených hodnot ze souboru, na ose X je datová stopa a na ose Y je rastr z hodnot přijatých dat. Názvy řad grafu vstupu jsou popsány v legendě.

  Typ grafu si můžeme zvolit pomocí funkce „chartPage.ChartType“. Vybral jsem graf „chartPage.ChartType = Excel.XlChartType.xlLineMarkers“. Jde o typ spojnicového grafu se značkami hodnot (obr 10.2.2).

```
chartPage.ChartType = Excel.XlChartType.; //změna typu grafu
```



Obr. 10.2.2 Různé typy grafu

Velikost grafu lze měnit ve funkci „Excel.ChartObject myChart“, změněním hodnot v závorce za příkazem Add. Třetí hodnota představuje šířku grafu a čtvrtá jeho výšku (obr. 10.2.3)

```
Excel.ChartObjects xlCharts = (Excel.ChartObjects)xlWorkSheet.ChartObjects(Type.Missing);
// Určení velikosti grafu
Excel.ChartObject myChart = (Excel.ChartObject)xlCharts.Add(100, 80, 650, 300);
Excel.Chart chartPage = myChart.Chart;

chartPage.SetSourceData(chartRange, Excel.XlRowCol.xlColumns);
```

Obr. 10.2.3 Část kódu pro měnění velikosti grafu

10.3 Aplikace pro přenos dat z terminálu

Přenos dat z terminálu se řeší za pomocí AT příkazů, jejichž účelem je jednodušší konfigurace modemu. Byl definován jednoduchý jazyk nazývaný AT-language - každý řádek začíná znaky AT (z anglického attention). Modem dokáže pracovat ve dvou režimech:

- v příkazovém, sloužícím k zadávání AT-příkazů modemu,
- v přenosovém, v němž mezi sebou modemy přenášejí data a nemohou přijímat AT-příkazy.

Modem nemůže pracovat v obou zároveň, ale může mezi nimi přepínat bez ztráty spojení. Přepínání se provádí tzv. escape sekvencí, která sestává ze tří stejných znaků stisknutých rychle po sobě.

Přehled základních AT-příkazů
AT-příkaz začíná dvojicí znaků AT a sestává z několika písmen a parametr je většinou číslo. Každý příkaz se ukončuje stiskem klávesy Enter. Na jeden řádek lze napsat i více příkazů - pak se počáteční

sekvence AT píše jen před prvním příkazem. Na jeden řádek je možno kromě prvních dvou znaků AT napsat maximálně 40 znaků. Pokud bude řádek delší, modem ohlásí chybu.

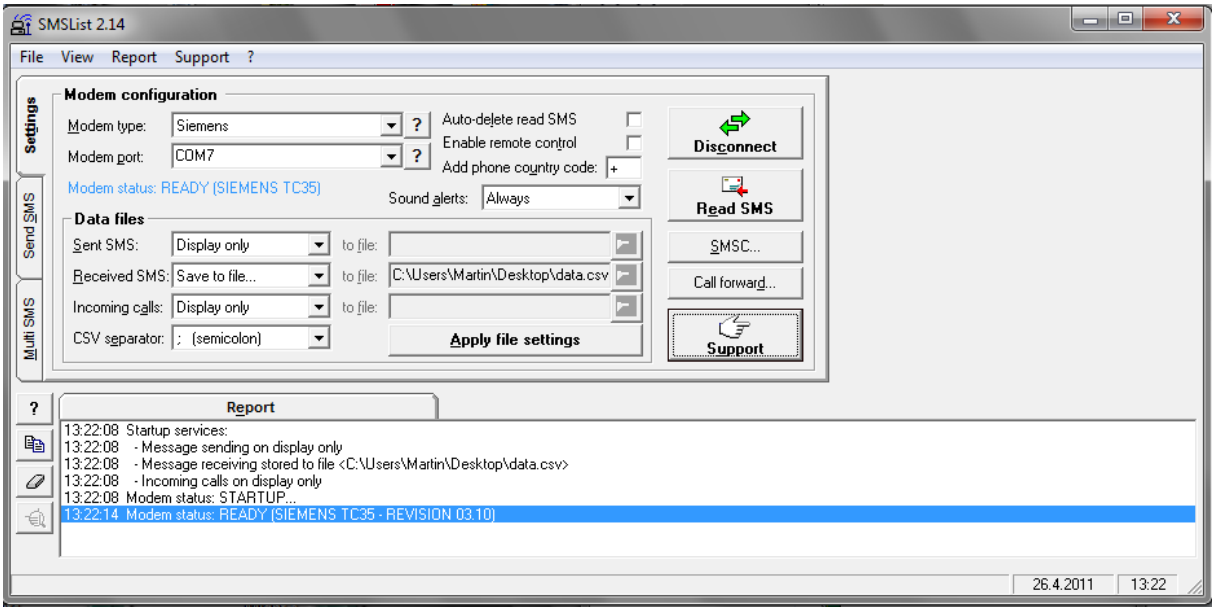
- **A/**
  Zopakování posledního platného příkazu. Kromě escape sekvence jediný AT-příkaz,který se nezadává s počátečním AT.
- **ATA**
  Odpověď na příchozí volání. Modem okamžitě zvedne linku a pokusí se navázat spojení.
- **AT+CMGL**
  Vypíše SMS zprávy.
- **AT+CMGD**
  Smazání příslušné SMS zprávy.
- **ATD**
  Vytočení telefonního čísla.
- **AT+CMGF=1**
  Nastavení formátu zprávy
- **AT+CSCA="XXX"**
  Nastavení čísla střediska zpráv XXX.
- **AT+CMGS="YYY"**
  Zadání čísla příjemce YYY.

10.3.1 SMSlist 2.14

Protože pro ovládání měřicí stanice potřebuji posílat SMS zprávy ve správných tvarech, například "POSLI" z terminálu, vybral jsem si pro použití vytvořenou aplikaci SMSlist 2.14, která je volně k dispozici na internetu. Tato aplikace umí posílat a přijímat SMS data z terminálu nebo mobilního telefonu, připojeného po sériové lince. Aplikace umožňuje ukládat přijaté SMS do textového souboru, nebo do souboru s příponou CSV.

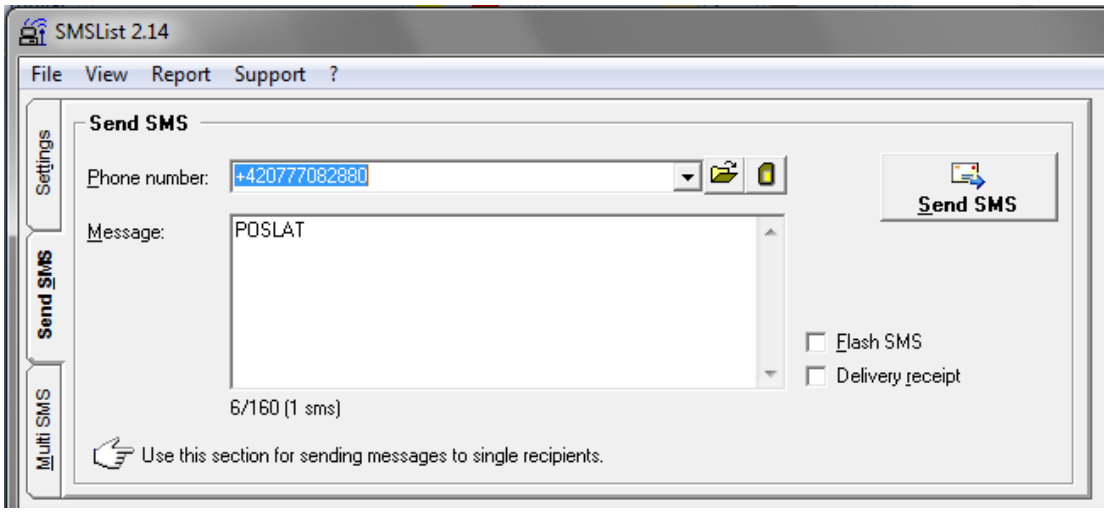
Po otevření aplikace je potřeba v záložce"settings" nastavit „Modem port" pro komunikaci. Dalším důležitým parametrem je v záložce „settings" v „Data files" nastavit pro ukládání přijatých SMS „recived SMS" pokyn „Save to File" a na stejném řádku nastavit cestu, kam chceme data ukládat.
Poté je potřeba uložit změny kliknutím na tlačítko „Apply file setings" a připojit se pomocí tlačítka conect (Obr. 10.3.1).

V textovém okně Report je vypsáno, zda se podařilo úspěšně připojit terminál, jeho status. Dále jsou zde zobrazeny přijaté a odeslané SMS.



Obr. 10.3.1 Nastaveni aplikace SMSlist pro připojeni

Odesílání SMS se provádí v záložce Send SMS, kde je potřeba nastavit číslo, kam chceme posílat SMS a její text, v mém případě je potřeba posílat text ve formátu „POSLAT“ (obr 10.3.2).

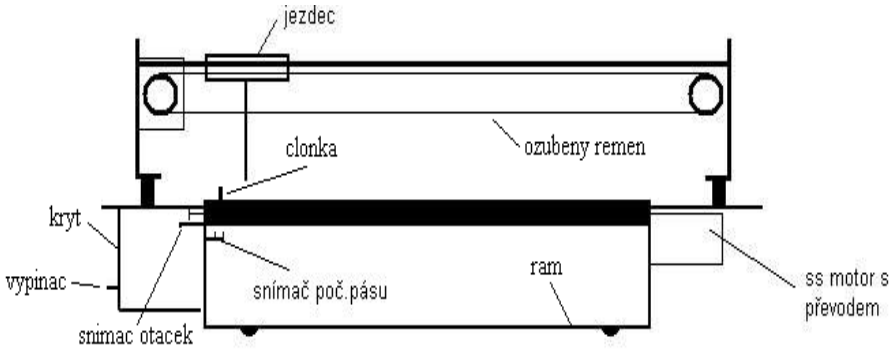


Obr. 10.3.2 Odesíláni SMS

11. Testovací měření

11.1. Testování digitálních kanálu a čítače

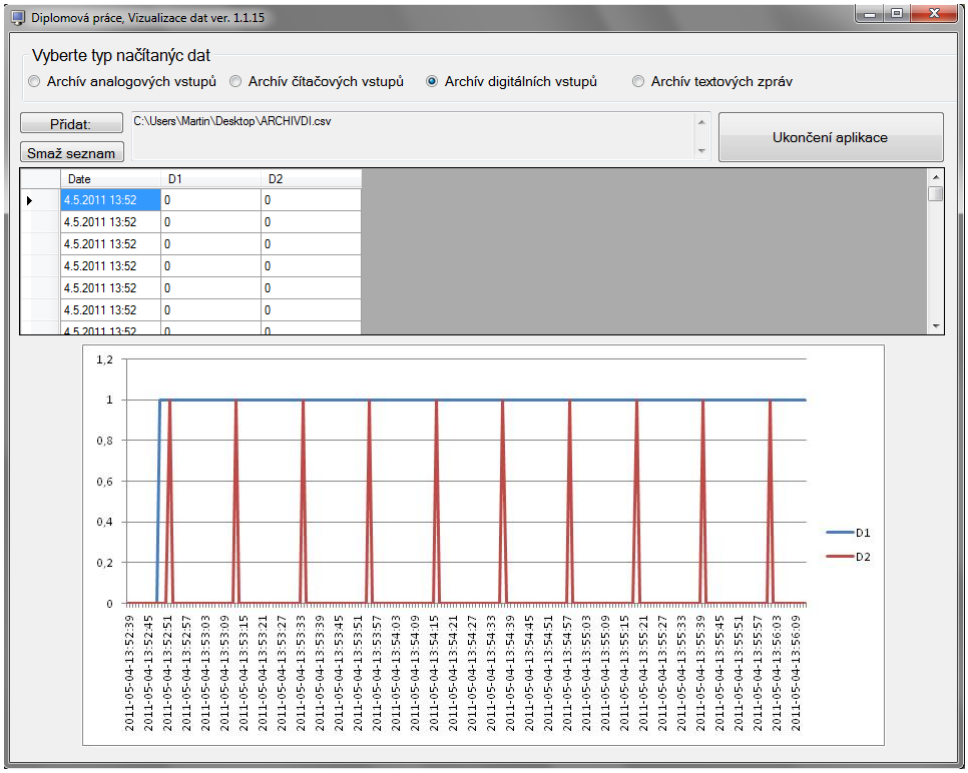
Pro demonstraci funkčnosti měřicí stanice byl použit přípravek polohovacího modulu EM253, který je umístěn ve školní laboratoři (NK317) a slouží pro výuku studentu. Proces je řízen pomocí PLC Simatic S7 200, který provádí veškeré potřebné operace pro to, aby se ukazatel vyhnul objektům postupujícím na páse. Úloha je založena na ovládání krokového motoru, který pohání přes gumový řemen pozici ukazatele a tak se vyhýbá překážkám. Pohyb krokového motoru je ovládán pomocí polohovacího modulu EM 253, který nabízí jednoduché nastavení a funkce pro jeho pohyb.



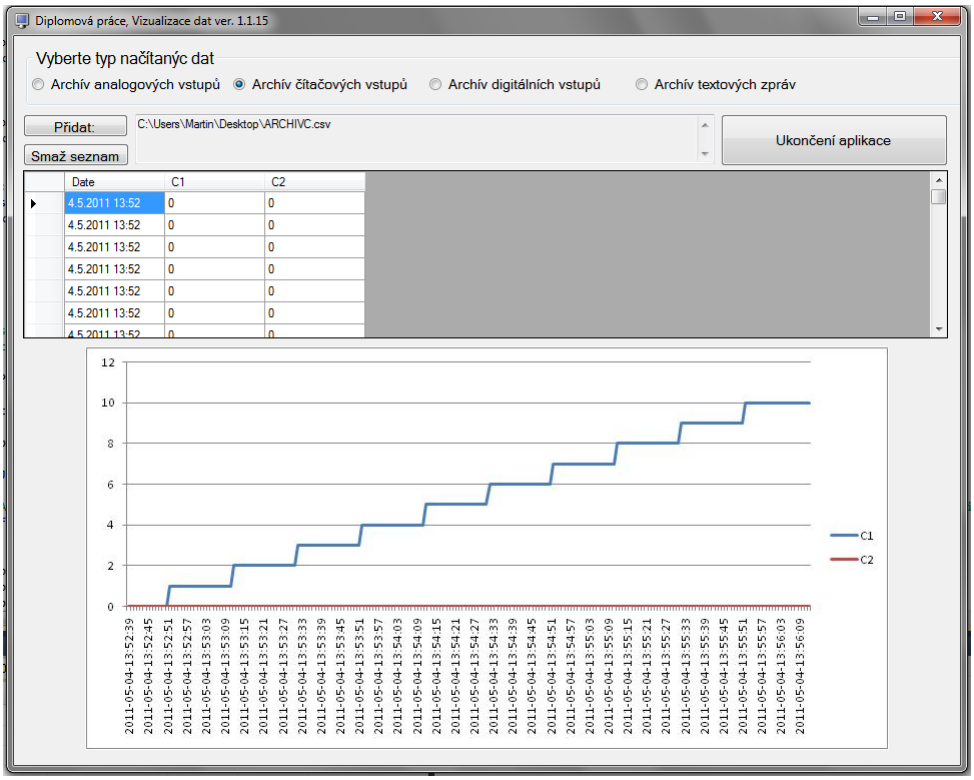
Obr. 11.1.1 Model běžícího pásu.

11.1.1 Měření a postup

Signály z přepínače pro zapnutí dopravního pásu byl připojen na první kanál K1 (DI1) měřicí stanice a na druhý kanál K2 (DI2) byl připojen signál snímače pro hledání referenčního bodu pásu. Signál počátku pásu byl připojen na čítačový vstup K3 (C1). Po zaslání řídicí SMS zprávy do měřicí stanice a zapnutí přípravku byla archivována naměřená data na paměťovou kartu. Po měření data byla data přenesena z paměti PLC do PC a výsledky graficky zpracovány. Naměřené časové průběhy jsou zobrazeny na obrázku 11.1.1.1, kde datová řada D1 zobrazuje stav zapnutí přípravku, datová řada D2 zobrazuje nalezení referenčního bodu pásu, který je důležitý pro správnou funkci přípravku. Obrázek 11.1.2 zobrazuje hodnoty čítače, kolikrát byl nalezen začátek pásu.



Obr. 11.1.1.1 Naměřená data digitálních vstupů



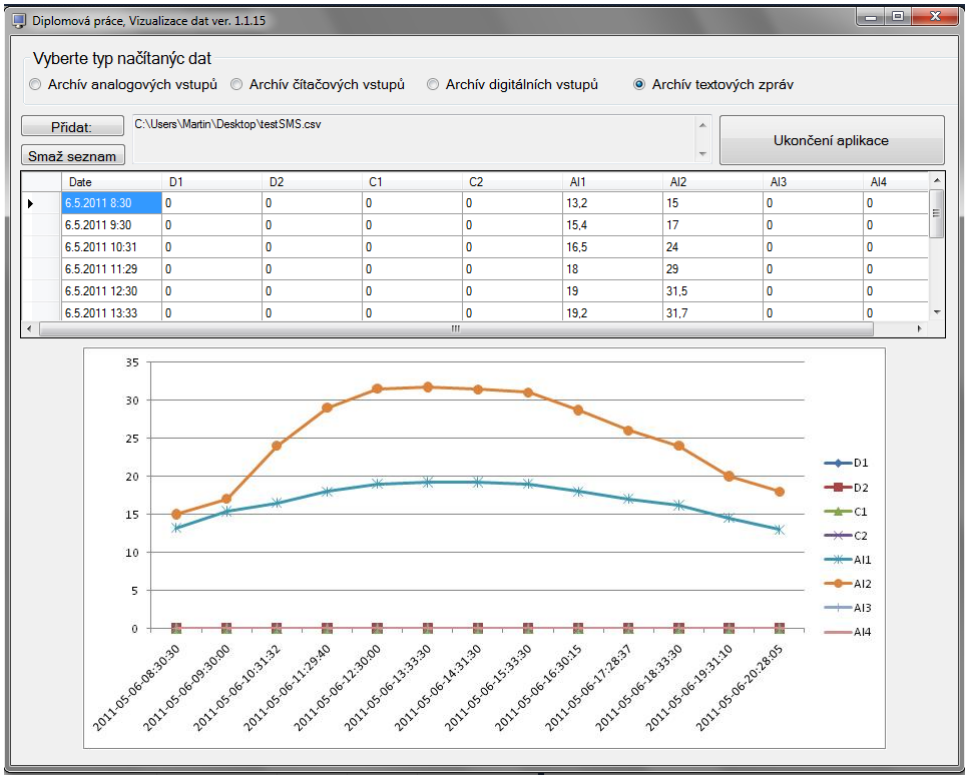
Obr. 11.1.1.2 Naměřená data čítačových vstupů

11.2. Testování analogových kanálů

K testování analogových vstupu bylo využito dvou teplotních čidel typu KTY 83-110 s maximálním teplotním měřicím rozsahem –55°C až +175 °C a tolerancí ±1%. Pro převod odporového signálu čidel na napětí jsou sestaveny dva převodníky R/U s operačním zesilovačem MA1458. Tím bylo získáno napětí téměř lineárně závislé na měřené teplotě.
Převod je nastaven takto: 0°C ≅ 0V
100°C ≅ 10V

11.2.1 Měření a postup

Pro měření bylo umístěné první teplotní čidlo do stínu a připojeno k měřicí stanici na první analogový kanál K5 (AI1), druhé bylo umístěno na slunci a připojeno na druhy analogový kanál K6 (AI2). Měření bylo prováděno každou hodinu od 8:30 do 20:30. Pomocí zasíláni řídící SMS zprávy s textem „POSLAT“ do měřicí stanice byla zpětně odeslána aktuální naměřena data. Tato data byla přijata terminálem a zpracována v počítači. Přijata data byla v aplikaci vynásobená hodnotou 10, tak aby na výstupu grafu byla uvedena teplota měřicích čidel. Řada AI2 (hnědá) je zobrazeni snímače umístěného na slunci, řada AI1(modrá) je zobrazení teploty snímače umístěného ve stínu.



Obr. 11.2.1.1.1 Grafické zpracování naměřených teplot snímačů

12. Závěr:

Diplomová práce se zabývá vytvořením ovládání měřicí stanice formou bezdrátového přenosu dat. Data jsou získáváná a archivována na vyžádání formou řídících SMS zpráv pomocí GSM brány, umístěné v měřicí stanici. Z mobilního zařízení do měřicí stanice jsou posílány řídící SMS zprávy, aktuální naměřená data jsou poté odeslána přes GSM bránu pomocí SMS zprávy do terminálu. Řídicí SMS zpráva pro příjem aktuálních naměřených dat musí obsahovat konkrétní příkaz jak nakládat s příchozím textem. Přijatá data z terminálu jsou uložena na počítač.

Součástí měřicí stanice je programovatelný automat. Při rešení úkolu byl použit datalogger typu TECOMAT FOXTROT firmy Teco, a.s. s centrální jednotkou CP 1004, která je doplněna paměťovou kartou o velikosti 1GB. Součástí měřicí stanice je GSM brána GSM2-01, svorkovnice a napájecí zdroj firmy ELKO EP, s.r.o. Svorkovnice slouží pro připojení vstupních signálů.

Ve své práci navazuji na diplomovou práci Ing. Jiřího Žáčka „Návrh a realizace autonomní stanice pro měření a archivaci dat“. Ve své práci řešil sběr dat v on-line režimu s nadřazeným systémem. Zdrojový kód programu Ing. Žáčka pro automat TECOMAT FOXTROT byl vytvořen v prvních verzích programovacího jazyka MOSAIC obsahujícího staré knihovny. Stará verze programovacího jazyka připouštěla nesprávné přiřazení typu hodnot a jejich nastavení.

V rámci této diplomové práce bylo nutno stávající program upravit tak, aby jej bylo možno přeložit a nahrát do programovatelného automatu TECOMAT FOXTROT, bylo rovněž zapotřebí změnit některé funkce programu a upravit zápis přiřazení hodnot, který byl v původním zdroji nesprávně zvolen. Pro správné fungování bezdrátového přenosu dat pomocí GMS brány a archivaci dat na paměťovou kartu bylo nutno nahrát do PLC nový Firmware, konkrétně Firmware FW_CP-1004_v65, který řeší správnou komunikaci s paměťovou kartou a GSM bránou. Protože GSM brána umožňuje odesílát pouze kratké textové zprávy a ne celková data pomoci GPRS, byl zdrojový kód programu doplněn o práci s paměťovou kartou. Na paměťovou kartu se data archivují a prenášejí se pomoci propojení měřicí stanice s počitačem. Všechny paměťové karty mají omezený počet zápisů, byla archivace dat řešena přesunem hodnot z databoxu do paměťové karty tak, aby karta nebyla zničena frekventovaným ukládáním aktuálních dat v každém cyklu programu. Data jsou archivována ve třech souborech typu CSV, podle druhů vstupnich kanálů. Zpracování přijatých a archivovaných dat je naprogramováno v programovacím jazyku C#. Získána data jsou uvedena ve formě tabulky a graficky zobrazena.

Při zkušebním testování dataloggeru TECOMAT FOXTROT bylo prováděno snímání teplot ze dvou teplotních čidel typu KTY 83-110 umístěnych ve stínu a na slunci. Zasíláním řídících SMS zpráv s textem „POSLAT“ do měřicí stanice byla formou SMS zpráv zpětně zasílána aktuální naměřená data. Data se zároveň ukládala na paměťovou kartu PLC. Řídící zprávy byly zasílány opakovaně v hodinových intervalech po dobu 12 hodin. Tato data byla přijatá terminálem, tabulkově a graficky zpracována.

Funkce měřicí stanice by mohla být v budoucnu rozšířena o GSM bránu, která by umožňovala odesílat nejen SMS zprávy ale také data přes GPRS. Archivovaná data z paměti PLC by se nemusela přenášet propojením měřicí stanice s počítačem.

Použitá literatura

[1] OMEGA - *Datalogery* [online]. 2009 [cit. 2011-05-03]. Dataloggery - přístroje pro záznam dat. Dostupné z WWW: <http://www.omegaeng.com/prodinfo/dataloggers.html>.

[2] SPŠ Rakovník. *Programovatelné automaty I* [online]. 25.02.2010 , 1. vydání, 1, [cit. 2011-05-03]. Dostupný z WWW: <http://www.spsrakovnik.cz/download.php?soubor=132>.

[3] *Teco, a.s. : Tecomat FOXTROT - nový, malý a modulární* [online]. 3. 5. 2010 [cit. 2011-05-03]. ŘÍDICÍ SYSTÉMY PRO STROJE, PROCESY A BUDOVY. Dostupné z WWW: <http://www.tecomat.cz/index2.php?lang=cs&m1id=1&m2id=4&m3id=11&mid=245>.

[4] Teco.a.s.. *Příručka projektanta systémů FOXTROT : Programovatelné automaty Tecomat FOXTROT* [online]. 03/2009, 13. vydání, TXV 004 10, [cit. 2011-05-03]. Dostupný z WWW: <http://www.tecomat.cz/docs/cze/Tecomat/txv00410.pdf>.

[5] Teco.a.s.. *Příručka projektanta systémů FOXTROT : Programování PLC TECOMAT podle IEC 61131-3* [online]. 03/2009, 3. vydání, TXV 004 11, [cit. 2011-05-03]. Dostupný z WWW: <http://www.tecomat.cz/docs/cze/tecomat/txv00411.pdf>.

[6] Teco.a.s.. *Příručka projektanta systémů FOXTROT : Příručka programátora PLC TECOMAT* [online]. 09/2007, 13. vydání, TXV 001 09, [cit. 2011-05-03]. Dostupný z WWW: < http://www.tecomat.cz/docs/cze/tecomat/txv00109.pdf >.

[7] *Teco, a.s. : Mosaic - vývojové prostředí vyhovující IEC* [online]. 15. 4. 2009 [cit. 2011-05-03]. ŘÍDICÍ SYSTÉMY PRO STROJE, PROCESY A BUDOVY. Dostupné z WWW: <http://www.tecomat.cz/index2.php?lang=cs&m1id=1&m2id=4&m3id=11&mid=245>.

[8] Teco.a.s.. *Příručka projektanta systémů FOXTROT : Knihovny pro programování PLC TECOMAT podle IEC 61131-3* [online]. 03/2006, 8. vydání, TXV 003 22, [cit. 2011-05-03]. Dostupný z WWW: <http://www.tecomat.cz/docs/cze/Software/Mosaic/TXV00322.pdf>.

[9] Teco.a.s.. *Příručka projektanta systémů FOXTROT : Knihovna GSMLib pro GSM* [online]. 09/2009, 5. vydání, TXV 003 40, [cit. 2011-05-03]. Dostupný z WWW: <http://www.tecomat.cz/docs/cze/Software/Mosaic/txv00340.pdf>.

[10] ŽÁČEK, Jiří. *Návrh a realizace autonomní stanice pro měření a archivaci dat*. Ostrava, 2008. 52 s s. Diplomová práce. VŠB – Technická univerzita Ostrava.

[11] Alphatech. *GSM modem Technické informace : GSM modem TC35* [online]. 10.4.2003 , 1. vydání, TXV 003 40, [cit. 2011-05-03]. Dostupný z WWW: <http://www.alphatech.cz/manualy/gsm_modem_tc35.pdf >.

Seznam příloh

Příloha 1.- Export zdrojového kódu z Tecomat FOXTROT

C:\MosaicApp\DiplomovaPrace\logger2\LOGGER2.ST

```
// kanály DI
TYPE
  typDI : STRUCT     // 15 B
    milisek  : USINT;
    sekunda  : USINT;
    minuta   : USINT;
    hodina   : USINT;
    den      : USINT;
    mesic    : USINT;
    rok      : USINT;
    hodnotaK1 : DINT;
    hodnotaK2 : DINT;
    END_STRUCT;
END_TYPE
//kanály čítačové C
TYPE
  typC : STRUCT       // 15 B
    milisek  : USINT;
    sekunda  : USINT;
    minuta   : USINT;
    hodina   : USINT;
    den      : USINT;
    mesic    : USINT;
    rok      : USINT;
    hodnotaK3 : DINT;
    hodnotaK4 : DINT;
    END_STRUCT;
END_TYPE
// kanály AI
TYPE
  typAI : STRUCT      // 23 B
    milisek  : USINT;
    sekunda  : USINT;
    minuta   : USINT;
    hodina   : USINT;
    den      : USINT;
    mesic    : USINT;
    rok      : USINT;
    hodnotaK5 : REAL;
    hodnotaK6 : REAL;
    hodnotaK7 : REAL;
    hodnotaK8 : REAL;
    END_STRUCT;
END_TYPE
VAR_GLOBAL RETAIN
  // K1,DI datová struktura
  fyzCisloK1       AT %R400   : INT    :=0;    //číslo fyzického kanálu
  logCisloK1       AT %R402   : INT    :=1;    //číslo logického kanálu
  nazevK1          AT %R404   : STRING :='digitalni 1'; //název kanálu
  jednotkaK1       AT %R486   : STRING :='digit'; //fyzikální jednotka
  aktivaceK1       AT %R568.0 : BOOL   :=TRUE;   //aktivace měření kanálu
  typVstupK1       AT %R569   : INT    :=1;//výběr typu vstupu
  negovatK1        AT %R571.0 : BOOL;    //negovat vstupní signál
  PVCitaceK1       AT %R572   : DINT;   //přednastavená hodnota čítače
  aktivaceSimulK1  AT %R576.0 : BOOL;   //povolení ruční simulované hodnoty
  hodnotaSimulK1   AT %R577   : DINT;   //zadaní ruční simulované hodnoty
  cisloArchivuK1   AT %R581   : INT;    //přiřazení archivu k tomuto kanalu
  aktivaceAlarmK1  AT %R583.0 : BOOL;   //aktivace alarmu
  zadaniHMAlarmK1  AT %R584   : DINT;   //zadání horní meze pro alarm
  zadaniDMAlarmK1  AT %R588   : DINT;   //zadání dolní meze pro alarm
  zadani2HMAlarmK1 AT %R592   : DINT;   //zadání druhé horní meze pro alarm
  zadani2DMAlarmK1 AT %R596   : DINT;   //zadání druhé dolní meze pro alarm
  zadaniUrAlarmK1  AT %R600.0 : BOOL;   //zadání log. úrovně digilál. signálu pro alarm
  reakceHMAlarmK1  AT %R601   : INT;    //zadání čísla reakce pro horní mez
  reakceDMAlarmK1  AT %R603   : INT;    //zadání čísla reakce pro dolní mez
  reakce2HMAlarmK1 AT %R605   : INT;    //zadání čísla reakce pro druhou horní mez
  reakce2DMAlarmK1 AT %R607   : INT;    //zadání čísla reakce pro druhou dolní mez
```

```
reakceUrAlarmK1  AT %R609   : INT;    //zadání čísla reakce pro úrovně digilál. signálu
cisloSvazAIK1    AT %R611   : INT;    //přiřazeni AI k svázanému DI
periodaVzK1      AT %R613   : INT;    //perioda vzorkování(pouze každý kanál zvlášť!)
startStopK1      AT %R615.0 : BOOL;   //start/stop (každý kanál zvlášť!!!)
// K2,DI datová struktura
fyzCisloK2       AT %R700   : INT    :=1;    //číslo fyzického kanálu
logCisloK2       AT %R702   : INT    :=2;    //číslo logického kanálu
nazevK2          AT %R704   : STRING :='digitalni 2'; //název kanálu
jednotkaK2       AT %R786   : STRING :='digit'; //fyzikální jednotka
aktivaceK2       AT %R868.0 : BOOL   :=TRUE;   //aktivace měření kanálu
typVstupK2       AT %R869   : INT    :=1;    //výběr typu vstupu
negovatK2        AT %R871.0 : BOOL;   //negovat vstupní signál
PVCitaceK2       AT %R872   : DINT;   //přednastavená hodnota čítače
aktivaceSimulK2  AT %R876.0 : BOOL;   //povolení ruční simulované hodnoty
hodnotaSimulK2   AT %R877   : DINT;   //zadaní ruční simulované hodnoty
cisloArchivuK2   AT %R881   : INT;    //přiřazení archivu k tomuto kanalu
aktivaceAlarmK2  AT %R883.0 : BOOL;   //aktivace alarmu
zadaniHMAlarmK2  AT %R884   : DINT;   //zadání horní meze pro alarm
zadaniDMAlarmK2  AT %R888   : DINT;   //zadání dolní meze pro alarm
zadani2HMAlarmK2 AT %R892   : DINT;   //zadání druhé horní meze pro alarm
zadani2DMAlarmK2 AT %R896   : DINT;   //zadání druhé dolní meze pro alarm
zadaniUrAlarmK2  AT %R900.0 : BOOL;   //zadání log. úrovně digilál. signálu pro alarm
reakceHMAlarmK2  AT %R901   : INT;    //zadání čísla reakce pro horní mez
reakceDMAlarmK2  AT %R903   : INT;    //zadání čísla reakce pro dolní mez
reakce2HMAlarmK2 AT %R905   : INT;    //zadání čísla reakce pro druhou horní mez
reakce2DMAlarmK2 AT %R907   : INT;    //zadání čísla reakce pro druhou dolní mez
reakceUrAlarmK2  AT %R909   : INT;    //zadání čísla reakce pro úrovně digilál. signálu
cisloSvazAIK2    AT %R911   : INT;    //přiřazeni AI k svázanému DI
periodaVzK2      AT %R913   : INT;    //perioda vzorkování (pouze každý kanál zvlášť!!!)
startStopK2      AT %R915.0 : BOOL;   //start/stop (pouze každý kanál zvlášť!!!)
  // K3,DI datová struktura
fyzCisloK3       AT %R1000   : INT     :=0;    //číslo fyzického kanálu
logCisloK3       AT %R1002   : INT     :=3;    //číslo logického kanálu
nazevK3          AT %R1004   : STRING  := 'čítač1'; //název kanálu
jednotkaK3       AT %R1086   : STRING  := 'impuls'; //fyzikální jednotka
aktivaceK3       AT %R1168.0 : BOOL    := TRUE;   //aktivace měření kanálu
typVstupK3       AT %R1169   : INT     :=3;//výběr typu vstupu
negovatK3        AT %R1171.0 : BOOL;   //negovat vstupní signál
PVCitaceK3       AT %R1172   : DINT;   //přednastavená hodnota čítače
aktivaceSimulK3  AT %R1176.0 : BOOL;   //povolení ruční simulované hodnoty
hodnotaSimulK3   AT %R1177   : DINT;   //zadaní ruční simulované hodnoty
cisloArchivuK3   AT %R1181   : INT;    //přiřazení archivu k tomuto kanalu
aktivaceAlarmK3  AT %R1183.0 : BOOL;   //aktivace alarmu
zadaniHMAlarmK3  AT %R1184   : DINT;   //zadání horní meze pro alarm
zadaniDMAlarmK3  AT %R1188   : DINT;   //zadání dolní meze pro alarm
zadani2HMAlarmK3 AT %R1192   : DINT;   //zadání druhé horní meze pro alarm
zadani2DMAlarmK3 AT %R1196   : DINT;   //zadání druhé dolní meze pro alarm
zadaniUrAlarmK3  AT %R1200.0 : BOOL;   //zadání log. úrovně digilál. signálu pro alarm
reakceHMAlarmK3  AT %R1201   : INT;    //zadání čísla reakce pro horní mez
reakceDMAlarmK3  AT %R1203   : INT;    //zadání čísla reakce pro dolní mez
reakce2HMAlarmK3 AT %R1205   : INT;    //zadání čísla reakce pro druhou horní mez
reakce2DMAlarmK3 AT %R1207   : INT;    //zadání čísla reakce pro druhou dolní mez
reakceUrAlarmK3  AT %R1209   : INT;    //zadání čísla reakce pro úrovně digilál. signálu
cisloSvazAIK3    AT %R1211   : INT;    //přiřazeni AI k svázanému DI
periodaVzK3      AT %R1213   : INT;    //perioda vzorkování (pouze každý kanál zvlášť!!!)
startStopK3      AT %R1215.0 : BOOL;   //start/stop (pouze každý kanálz vlášť!!!)
  // K4,DI datová struktura
fyzCisloK4       AT %R1300   : INT    :=1;    //číslo fyzického kanálu
logCisloK4       AT %R1302   : INT    :=4;    //číslo logického kanálu
nazevK4          AT %R1304   : STRING :='čítač 2'; //název kanálu
jednotkaK4       AT %R1386   : STRING :='impuls'; //fyzikální jednotka
aktivaceK4       AT %R1468.0 : BOOL   :=TRUE;   //aktivace měření kanálu
typVstupK4       AT %R1469   : INT    :=3;    //výběr typu vstupu
negovatK4        AT %R1471.0 : BOOL;   //negovat vstupní signál
PVCitaceK4       AT %R1472   : DINT;   //přednastavená hodnota čítače
aktivaceSimulK4  AT %R1476.0 : BOOL;   //povolení ruční simulované hodnoty
hodnotaSimulK4   AT %R1477   : DINT;   //zadaní ruční simulované hodnoty
cisloArchivuK4   AT %R1481   : INT;    //přiřazení archivu k tomuto kanalu
aktivaceAlarmK4  AT %R1483.0 : BOOL;   //aktivace alarmu
zadaniHMAlarmK4  AT %R1484   : DINT;   //zadání horní meze pro alarm
```

```
zadaniDMAlarmK4  AT %R1488   : DINT;   //zadání dolní meze pro alarm
zadani2HMAlarmK4 AT %R1492   : DINT;   //zadání druhé horní meze pro alarm
zadani2DMAlarmK4 AT %R1496   : DINT;   //zadání druhé dolní meze pro alarm
zadaniUrAlarmK4  AT %R1500.0 : BOOL;   //zadání log. úrovně digilál. signálu pro alarm
reakceHMAlarmK4  AT %R1501   : INT;    //zadání čísla reakce pro horní mez
reakceDMAlarmK4  AT %R1503   : INT;    //zadání čísla reakce pro dolní mez
reakce2HMAlarmK4 AT %R1505   : INT;    //zadání čísla reakce pro druhou horní mez
reakce2DMAlarmK4 AT %R1507   : INT;    //zadání čísla reakce pro druhou dolní mez
reakceUrAlarmK4  AT %R1509   : INT;    //zadání čísla reakce pro úrovně digilál. signálu
cisloSvazAIK4    AT %R1511   : INT;    //přiřazeni AI k svázanému DI
periodaVzK4      AT %R1513   : INT;    //perioda vzorkování (pouze každý kanál zvlášť!!!)
startStopK4      AT %R1515.0 : BOOL;   //start/stop (pouze pro každý kanál zvlášť!!!)
// K5,AI datová struktura
fyzCisloK5       AT %R1600   : INT     :=0;    //číslo fyzického kanálu
logCisloK5       AT %R1602   : INT     :=5;    //číslo logického kanálu
nazevK5          AT %R1604   : STRING  :='analogový 1'; //název kanálu
jednotkaK5       AT %R1686   : STRING  :='[V]'; //fyzikální jednotka
aktivaceK5       AT %R1768.0 : BOOL    :=TRUE;   //aktivace měření kanálu
typDatK5         AT %R1769   : INT     :=3;    //výběr reprezentace dat ADC
elRozsahHMK5     AT %R1771   : REAL    :=10.0;   //zadání horní meze rozsahu el. signálu
elRozsahDMK5     AT %R1775   : REAL    :=0.0;   //zadání dolní meze rozsahu el. signálu
techRozsahHMK5   AT %R1779   : REAL    :=10.0;//zadání horní meze rozsahu techn. signálu
techRozsahDMK5   AT %R1783   : REAL    :=0.0;   //zadání dolní meze rozsahu tech. signálu
aktivaceSimulK5  AT %R1787.0 : BOOL;   //povolení ruční simulované hodnoty
hodnotaSimulK5   AT %R1788   : REAL;   //zadaní ruční simulované hodnoty
cisloArchivuK5   AT %R1792   : INT;    //přiřazení archivu k tomuto kanalu
aktivaceAlarmK5  AT %R1794.0 : BOOL;   //aktivace alarmu
zadaniHMAlarmK5  AT %R1795   : REAL;   //zadání horní meze pro alarm
zadaniDMAlarmK5  AT %R1799   : REAL;   //zadání dolní meze pro alarm
zadani2HMAlarmK5 AT %R1803   : REAL;   //zadání druhé horní meze pro alarm
zadani2DMAlarmK5 AT %R1807   : REAL;   //zadání druhé dolní meze pro alarm
reakceHMAlarmK5  AT %R1811   : INT;    //zadání čísla reakce pro horní mez
reakceDMAlarmK5  AT %R1813   : INT;    //zadání čísla reakce pro dolní mez
reakce2HMAlarmK5 AT %R1815   : INT;    //zadání čísla reakce pro druhou horní mez
reakce2DMAlarmK5 AT %R1817   : INT;    //zadání čísla reakce pro druhou dolní mez
aktivacePrumerK5 AT %R1819.0 : BOOL;   //aktivace funkce průměrování
intervalPrumerK5 AT %R1820   : INT;    //zadání čas. intervalu ve kterém se počítá průměr
filtraceK5       AT %R1822   : INT;    //filtrace signálu: 0 = zákaz filtrace, >0 = způsob
periodaVzK5      AT %R1824   : INT;    //perioda vzorkování (pouze každý kanál zvlášť!!!)
startStopK5      AT %R1826.0 : BOOL;   //start/stop (pouze každý kanál zvlášť!!!)
// K6,AI datová struktura
fyzCisloK6       AT %R1900   : INT    :=1;    //číslo fyzického kanálu
logCisloK6       AT %R1902   : INT    :=6;    //číslo logického kanálu
nazevK6          AT %R1904   : STRING :='analogový 2'; //název kanálu
jednotkaK6       AT %R1986   : STRING :='[V]'; //fyzikální jednotka
aktivaceK6       AT %R2068.0 : BOOL   :=TRUE;   //aktivace měření kanálu
typDatK6         AT %R2069   : INT    :=3;    //výběr reprezentace dat ADC
elRozsahHMK6     AT %R2071   : REAL   :=10.0;   //zadání horní meze rozsahu el. signálu
elRozsahDMK6     AT %R2075   : REAL   :=0.0;   //zadání dolní meze rozsahu el. signálu
techRozsahHMK6   AT %R2079   : REAL   :=10.0;   //zadání horní meze rozsahu tech.signálu
techRozsahDMK6   AT %R2083   : REAL   :=0.0;   //zadání dolní meze rozsahu tech. signálu
aktivaceSimulK6  AT %R2087.0 : BOOL;   //povolení ruční simulované hodnoty
hodnotaSimulK6   AT %R2088   : REAL;   //zadaní ruční simulované hodnoty
cisloArchivuK6   AT %R2092   : INT;    //přiřazení archivu k tomuto kanalu
aktivaceAlarmK6  AT %R2094.0 : BOOL;   //aktivace alarmu
zadaniHMAlarmK6  AT %R2095   : REAL;   //zadání horní meze pro alarm
zadaniDMAlarmK6  AT %R2099   : REAL;   //zadání dolní meze pro alarm
zadani2HMAlarmK6 AT %R2103   : REAL;   //zadání druhé horní meze pro alarm
zadani2DMAlarmK6 AT %R2107   : REAL;   //zadání druhé dolní meze pro alarm
reakceHMAlarmK6  AT %R2111   : INT;    //zadání čísla reakce pro horní mez
reakceDMAlarmK6  AT %R2113   : INT;    //zadání čísla reakce pro dolní mez
reakce2HMAlarmK6 AT %R2115   : INT;    //zadání čísla reakce pro druhou horní mez
reakce2DMAlarmK6 AT %R2117   : INT;    //zadání čísla reakce pro druhou dolní mez
aktivacePrumerK6 AT %R2119.0 : BOOL;   //aktivace funkce průměrování
intervalPrumerK6 AT %R2120   : INT;    //zadání čas. intervalu ve kterém se počítá průměr
filtraceK6       AT %R2122   : INT;    //filtrace signálu: 0 = zákaz filtrace, >0 = způsob
periodaVzK6      AT %R2124   : INT;    //perioda vzorkování (pouze každý kanál zvlášť!!!)
startStopK6      AT %R2126.0 : BOOL;   //start/stop (pouze každý kanál zvlášť!!!)
// K7,AI datová struktura
fyzCisloK7       AT %R2200   : INT    :=2;    //číslo fyzického kanálu
```

```
  logCisloK7       AT %R2202   : INT    :=7;    //číslo logického kanálu
  nazevK7          AT %R2204   : STRING :='analogový 3'; //název kanálu
  jednotkaK7       AT %R2286   : STRING :='[V]'; //fyzikální jednotka
  aktivaceK7       AT %R2368.0 : BOOL   :=TRUE;   //aktivace měření kanálu
  typDatK7         AT %R2369   : INT    :=3;    //výběr reprezentace dat ADC
  elRozsahHMK7     AT %R2371   : REAL   :=10.0;   //zadání horní meze rozsahu el. signálu
  elRozsahDMK7     AT %R2375   : REAL   :=0.0;   //zadání dolní meze rozsahu el. signálu
  techRozsahHMK7   AT %R2379   : REAL   :=10.0;   //zadání horní meze rozsahu tech. signálu
  techRozsahDMK7   AT %R2383   : REAL   :=0.0;   //zadání dolní meze rozsahu tech. signálu
  aktivaceSimulK7  AT %R2387.0 : BOOL;   //povolení ruční simulované hodnoty
  hodnotaSimulK7   AT %R2388   : REAL;   //zadaní ruční simulované hodnoty
  cisloArchivuK7   AT %R2392   : INT;    //přiřazení archivu k tomuto kanalu
  aktivaceAlarmK7  AT %R2394.0 : BOOL;   //aktivace alarmu
  zadaniHMAlarmK7  AT %R2395   : REAL;   //zadání horní meze pro alarm
  zadaniDMAlarmK7  AT %R2399   : REAL;   //zadání dolní meze pro alarm
  zadani2HMAlarmK7 AT %R2403   : REAL;   //zadání druhé horní meze pro alarm
  zadani2DMAlarmK7 AT %R2407   : REAL;   //zadání druhé dolní meze pro alarm
  reakceHMAlarmK7  AT %R2411   : INT;    //zadání čísla reakce pro horní mez
  reakceDMAlarmK7  AT %R2413   : INT;    //zadání čísla reakce pro dolní mez
  reakce2HMAlarmK7 AT %R2415   : INT;    //zadání čísla reakce pro druhou horní mez
  reakce2DMAlarmK7 AT %R2417   : INT;    //zadání čísla reakce pro druhou dolní mez
  aktivacePrumerK7 AT %R2419.0 : BOOL;   //aktivace funkce průměrování
  intervalPrumerK7 AT %R2420   : INT;    //zadání čas. intervalu ve kterém se počítá průměr
  filtraceK7       AT %R2422   : INT;    //filtrace signálu: 0 = zákaz filtrace, >0 = způsob
periodaVzK7      AT %R2424   : INT;    //perioda vzorkování (pouze každý kanál zvlášť!!!)
  startStopK7      AT %R2426.0 : BOOL;   //start/stop (pouze každý kanál zvlášť!!!)
  // K8,AI datová struktura
  fyzCisloK8       AT %R2500   : INT    :=3;    //číslo fyzického kanálu
  logCisloK8       AT %R2502   : INT    :=8;    //číslo logického kanálu
  nazevK8          AT %R2504   : STRING :='analogový 4'; //název kanálu
  jednotkaK8       AT %R2586   : STRING :='[V]'; //fyzikální jednotka
  aktivaceK8       AT %R2668.0 : BOOL   :=TRUE;   //aktivace měření kanálu
  typDatK8         AT %R2669   : INT    :=3;    //výběr reprezentace dat ADC
  elRozsahHMK8     AT %R2671   : REAL   :=10.0;   //zadání horní meze rozsahu el. signálu
  elRozsahDMK8     AT %R2675   : REAL   :=0.0;   //zadání dolní meze rozsahu el. signálu
  techRozsahHMK8   AT %R2679   : REAL   :=10.0;   //zadání horní meze rozsahu tech. signálu
  techRozsahDMK8   AT %R2683   : REAL   :=0.0;   //zadání dolní meze rozsahu tech. signálu
  aktivaceSimulK8  AT %R2687.0 : BOOL;   //povolení ruční simulované hodnoty
  hodnotaSimulK8   AT %R2688   : REAL;   //zadaní ruční simulované hodnoty
  cisloArchivuK8   AT %R2692   : INT;    //přiřazení archivu k tomuto kanalu
  aktivaceAlarmK8  AT %R2694.0 : BOOL;   //aktivace alarmu
  zadaniHMAlarmK8  AT %R2695   : REAL;   //zadání horní meze pro alarm
  zadaniDMAlarmK8  AT %R2699   : REAL;   //zadání dolní meze pro alarm
  zadani2HMAlarmK8 AT %R2703   : REAL;   //zadání druhé horní meze pro alarm
  zadani2DMAlarmK8 AT %R2707   : REAL;   //zadání druhé dolní meze pro alarm
  reakceHMAlarmK8  AT %R2711   : INT;    //zadání čísla reakce pro horní mez
  reakceDMAlarmK8  AT %R2713   : INT;    //zadání čísla reakce pro dolní mez
  reakce2HMAlarmK8 AT %R2715   : INT;    //zadání čísla reakce pro druhou horní mez
  reakce2DMAlarmK8 AT %R2717   : INT;    //zadání čísla reakce pro druhou dolní mez
  aktivacePrumerK8 AT %R2719.0 : BOOL;   //aktivace funkce průměrování
  intervalPrumerK8 AT %R2720   : INT;    //zadání čas. intervalu ve kterém se počítá průměr
  filtraceK8       AT %R2722   : INT;    //filtrace signálu: 0 = zákaz filtrace, >0 = způsob
  periodaVzK8      AT %R2724   : INT;    //perioda vzorkování (pouze každý kanál zvlášť!!!)
  startStopK8      AT %R2726.0 : BOOL;   //start/stop (pouze každý kanál zvlášť!!!)
  //ovládání start stop měření a perioda vzorkování pro DI (K1, K2)
  periodaVzDI         AT %R2800   : INT  := 1;  // perioda vzorkování
  rucniStartStopDI    AT %R2802.0 : BOOL ; //ruční nadřazený start/stop měření
  aktivCasStartStopDI AT %R2802.1 : BOOL; //aktivace spuštění měření od realného času
  casStartSekDI       AT %R2803   : INT;  //čas a datum spuštění
  casStartMinDI       AT %R2805   : INT;
  casStartHodDI       AT %R2807   : INT;
  casStartDenDI       AT %R2809   : INT;
  casStartMesDI       AT %R2811   : INT;
  casStartRokDI       AT %R2813   : INT;
  casStopSekDI        AT %R2815   : INT;  //čas a datum stopnutí
  casStopMinDI        AT %R2817   : INT;
  casStopHodDI        AT %R2819   : INT;
  casStopDenDI        AT %R2821   : INT;
  casStopMesDI        AT %R2823   : INT;
  casStopRokDI        AT %R2825   : INT;
```

```
 aktivDIStartStopDI  AT %R2827.0 : BOOL; //aktivace spuštění měření od DI svázaného s AI
 metodaDIStartStopDI AT %R2829   : INT;  //metoda pro start a stop od DI svázaného s AI
//ovládání start stop měření a perioda vzorkování pro čítače C (K3, K4)
 periodaVzC         AT %R2850   : INT   := 1;   // perioda vzorkování
 rucniStartStopC    AT %R2852.0 : BOOL;  //ruční nadřazený start/stop měření
 aktivCasStartStopC AT %R2852.1 : BOOL;  //aktivace spuštění měření od realného času
 casStartSekC       AT %R2853   : INT;   //čas a datum spuštění
 casStartMinC       AT %R2855   : INT;
 casStartHodC       AT %R2857   : INT;
 casStartDenC       AT %R2859   : INT;
 casStartMesC       AT %R2861   : INT;
 casStartRokC       AT %R2863   : INT;
 casStopSekC        AT %R2865   : INT;   //čas a datum stopnutí
 casStopMinC        AT %R2867   : INT;
 casStopHodC        AT %R2869   : INT;
 casStopDenC        AT %R2871   : INT;
 casStopMesC        AT %R2873   : INT;
 casStopRokC        AT %R2875   : INT;
 aktivDIStartStopC  AT %R2877.0 : BOOL;  //aktivace spuštění měření od DI svázaného s AI
 metodaDIStartStopC AT %R2879   : INT;   //metoda pro start a stop od DI svázaného s AI
 //ovládání start stop měření a perioda vzorkování pro AI (K5, K6, K7, K8)
 periodaVzAI         AT %R2900   : INT  := 1;   // perioda vzorkování
 rucniStartStopAI    AT %R2902.0 : BOOL;  //ruční nadřazený start/stop měření
 aktivCasStartStopAI AT %R2902.1 : BOOL;  //aktivace spuštění měření od realného času
 casStartSekAI       AT %R2903   : INT;   //čas a datum spuštění
 casStartMinAI       AT %R2905   : INT;
 casStartHodAI       AT %R2907   : INT;
 casStartDenAI       AT %R2909   : INT;
 casStartMesAI       AT %R2911   : INT;
 casStartRokAI       AT %R2913   : INT;
 casStopSekAI        AT %R2915   : INT;   //čas a datum stopnutí
 casStopMinAI        AT %R2917   : INT;
 casStopHodAI        AT %R2919   : INT;
 casStopDenAI        AT %R2921   : INT;
 casStopMesAI        AT %R2923   : INT;
 casStopRokAI        AT %R2925   : INT;
 aktivDIStartStopAI  AT %R2927.0 : BOOL;  //aktivace spuštění měření od DI svázaného s AI
 metodaDIStartStopAI AT %R2929   : INT;   //metoda pro start a stop od DI svázaného s AI
 //archív DI
 cisloArchivuDI   AT %R2950 : INT  :=1;     //identifikační číslo archívu
 zacatekArchivuDI AT %R2952 : DINT  :=0;    //počáteční adresa archívu v paměti
 delkaArchivuDI   AT %R2956 : DINT  :=150000;    //délka archívu puvodne 150000
 delkaVetyDI      AT %R2960 : DINT;    //délka jednoho záznamu v archívu
 ukazatelDI       AT %R2964 : DINT;    //ukazatel na aktuální pozici v archívu
 loggDI           AT %R2968 : typDI;  // jeden logg (vzorek) daný strukturou
 realArchivDI     AT %R2983 : typDI;  // přepisovatelný archív reálných dat tvořen strukturou
 //archív čítače C
 cisloArchivuC   AT %R3000 : INT  :=1;   //identifikační číslo archívu
 zacatekArchivuC AT %R3002 : DINT  :=150000;   //počáteční adresa archívu v paměti
 delkaArchivuC   AT %R3006 : DINT  :=150000;   //délka archívu puvodne 150000
 delkaVetyC      AT %R3010 : DINT;   //délka jednoho záznamu v archívu
 ukazatelC       AT %R3014 : DINT;   //ukazatel na aktuální pozici v archívu
 loggC           AT %R3018 : typC;  // jeden logg (vzorek) daný strukturou
 realArchivC     AT %R3033 : typC;  // přepisovatelný archív reálných dat tvořen strukturou
 //archív AI
 cisloArchivuAI   AT %R3050 : INT  :=1;    //identifikační číslo archívu
 zacatekArchivuAI AT %R3052 : DINT  :=300000;    //počáteční adresa archívu v paměti
 delkaArchivuAI   AT %R3056 : DINT  :=212000;    //délka archívu  puvodne 212000
 delkaVetyAI      AT %R3060 : DINT;    //délka jednoho záznamu v archívu
 ukazatelAI       AT %R3064 : DINT;    //ukazatel na aktuální pozici v archívu
 loggAI           AT %R3068 : typAI;  // jeden logg (vzorek) daný strukturou
 realArchivAI     AT %R3091 : typAI;  // přepisovatelný archív reálných dat tvořen strukturou
  // globální nastavovací prvky
 restartDataloggeru AT %R3120 : INT;        // proměnná pro studený restart dataloggeru
 aktivaceSMS        AT %R3122.0 : BOOL;       //povolení zasílíání SMS
 telCisloSMS        AT %R3123 : STRING[20] :='+420XXXXXXXXX'; // tel. číslo pro příjem SMS
 // stahování archívů DI
 vycAddressDI AT %R3200 : DINT;      // počáteční adresa věty pro vyčtení
 vycLengthDI  AT %R3204 : INT;       // délka věty pro vyčtení
 vycControlDI AT %R3206 : INT;       // řízení vyčítání 0-neděje se nic, 1-vyčíst novou větu
```

```
vycBufferDI  AT %R3208 : typDI;     // buffer vyčtených dat
promoticPointerDI AT %R3223 : DINT;  promPocitadloDI AT %R3227 : DINT;
straceneZaznamyDI AT %R3231 : DINT; // počet stracených vět při výpadku vyčítání dat z PLC
// stahování archívů C
vycAddressC AT %R3250 : DINT;      // počáteční adresa věty pro vyčtení
vycLengthC  AT %R3254 : INT;       // délka věty pro vyčtení
vycControlC AT %R3256 : INT;       // řízení vyčítání 0-neděje se nic, 1-vyčíst novou větu
vycBufferC  AT %R3258 : typC;      // buffer vyčtených dat
promoticPointerC AT %R3273 : DINT;  promPocitadloC  AT %R3277 : DINT;  /
straceneZaznamyC AT %R3281 : DINT;//počet ztracených vět při výpadku vyčítání dat z paměti
  // stahování archívů AI
vycAddressAI AT %R3300 : DINT;      // počáteční adresa věty pro vyčtení
vycLengthAI  AT %R3304 : INT;       // délka věty pro vyčtení
vycControlAI AT %R3306 : INT;       // řízení vyčítání 0-neděje se nic, 1-vyčíst novou větu
vycBufferAI  AT %R3308 : typAI;     // buffer vyčtených dat
promoticPointerAI AT %R3331 : DINT;  promPocitadloAI AT %R3335 : DINT;   /
straceneZaznamyAI AT %R3339 : DINT; // počet stracených vět při výpadku vyčítání dat
////////////////////////////////////dalši vnitřní proměnné//////////////////////////////
misek AT %S5 : USINT;  //  ukazatel na RTC
sek  AT %S6  : USINT;
minut AT %S7 : USINT;
hod  AT %S8  : USINT;
den  AT %S10 : USINT;
mes  AT %S11 : USINT;
rok  AT %S12 : USINT;
casovacDI : BOOL;   //promenna slouzi pro podminku k casovani loggovani
casovacC : BOOL;    // ukazuje na jednu z nasledujicich promennych
casovacAI : BOOL;
ms100 AT %S20.0 : BOOL;   //ukazatele na nabezne hrany casovych jednotek
ms500 AT %S20.1 : BOOL;
s1 AT %S20.2 : BOOL;
s10 AT %S20.3 : BOOL;
minut1 AT %S20.4 : BOOL;
minut10 AT %S20.5 : BOOL;
hod1 AT %S20.6 : BOOL;
den1 AT %S20.7 : BOOL;
runDI : BOOL:= true;  // promenna pro podminku spusteni mereni DI
runC : BOOL:= true;   // promenna pro podminku spusteni mereni DI
runAI : BOOL := true;  // promenna pro podminku spusteni mereni DI
valK1 : BOOL; // obraz přiřazeného fyz. vstupu
valK2 : BOOL; // obraz přiřazeného fyz. vstupu
valK3 : DINT; // obraz přiřezeného čítače
valK4 : DINT; // obraz přiřazeného čítače
valK5 : REAL; // obraz přiřazeného fyz. vstupu
valK6 : REAL; // obraz přiřazeného fyz. vstupu
valK7 : REAL; // obraz přiřazeného fyz. vstupu
valK8 : REAL; // obraz přiřazeného fyz. vstupu
valueK1 : BOOL; // obraz vstupu po podmínce negace
valueK2 : BOOL; // obraz vstupu po podmínce negace
valueK5 : REAL; // obraz vstupu po přepočítání na uživ. rozměr
valueK6 : REAL; // obraz vstupu po přepočítání na uživ. rozměr
valueK7 : REAL; // obraz vstupu po přepočítání na uživ. rozměr
valueK8 : REAL; // obraz vstupu po přepočítání na uživ. rozměr
convertValueK1 : DINT; // obraz vstupu DI (po podmínce negace a převodu na DINT)
convertValueK2 : DINT; // obraz vstupu DI (po podmínce negace a převodu na DINT)
convertValueK3 : DINT; // obraz valK3
convertValueK4 : DINT; // obraz valK4
enableCnt : BOOL := TRUE;    //povoleni citacu kanal K3 a K4
citA   : UDINT;      //hodnota čítače A (r0_p3_CNT_IN2.VALA)
citAA  : UDINT;      //hodnota čítače A v předchozím cyklu programu
difA   : UDINT;      //změna hodnoty čítače v cyklu programu
convertDifA : DINT;  //convetr předchozí proměnné
citB   : UDINT;      //hodnota čítače B (r0_p3_CNT_IN2.VALB)
citBB  : UDINT;      //hodnota čítače B v předchozím cyklu programu
difB   : UDINT;      //změna hodnoty čítače v cyklu programu
convertDifB : DINT;  //convetr předchozí proměnné
pvCntK3 : DINT;   // obraz proměnné PVCitaceK3
pvCntK4 : DINT;   // obraz proměnné PVCitaceK4
pvCntKK3 : DINT;  // obraz proměnné PVCitaceK3 (pomocná pro určení změny hodnoty)
pvCntKK4 : DINT;  // obraz proměnné PVCitaceK4 (pomocná pro určení změny hodnoty)
```

```
koefK5 : REAL;  //koeficient k v rovnici přímky k*x+q
koefK6 : REAL;  //koeficient k v rovnici přímky k*x+q
koefK7 : REAL;  //koeficient k v rovnici přímky k*x+q
koefK8 : REAL;  //koeficient k v rovnici přímky k*x+q
pocitadloDI AT %R3424 : DINT;  //počítadlo počtu přepisů archívu DI
pocitadloC  AT %R3428 : DINT;  //počítadlo počtu přepisů archívu C
pocitadloAI AT %R3432 : DINT;  //počítadlo počtu přepisů archívu AI
offsetDI : DINT ;  // ukazatel na aktuální pozici kam chci ukládat
offsetC  : DINT ;  // ukazatel na aktuální pozici kam chci ukládat
offsetAI : DINT ;  // ukazatel na aktuální pozici kam chci ukládat
hodnotaNeaktivDIC : DINT := 99999;  // archivovaná hodnota neaktivních kanálů DI a C
hodnotaNeaktivAI  : REAL := 99999;  // archivovaná hodnota neaktivních kanálů AI
zacatek1DI    : DINT;   // pomocná pro určení změny začátku archívu
zacatek2DI    : DINT;   // druhá pomocná pro určení změny začátku archívu
delka1DI      : DINT;   // pomocná pro určení změny délky archívu
delka2DI      : DINT;   // druhá pomocná pro určení změny délky archívu
zmenaDI       : BOOL;   // indikace změny, pokud nastala změna začátku nebo délky
zacatek1C     : DINT;   // pomocná pro určení změny začátku archívu
zacatek2C     : DINT;   // druhá pomocná pro určení změny začátku archívu
delka1C       : DINT;   // pomocná pro určení změny délky archívu
delka2C       : DINT;   // druhá pomocná pro určení změny délky archívu
zmenaC        : BOOL;   // indikace změny, pokud nastala změna začátku nebo délky
zacatek1AI    : DINT;   // pomocná pro určení změny začátku archívu
zacatek2AI    : DINT;   // druhá pomocná pro určení změny začátku archívu
delka1AI      : DINT;   // pomocná pro určení změny délky archívu
delka2AI      : DINT;   // druhá pomocná pro určení změny délky archívu
zmenaAI       : BOOL;   // indikace změny, pokud nastala změna začátku nebo délky
zacatekPracDI AT %R3400 : DINT;   // pracovní ekvivalent proměnné zacatekArchivuDI
delkaPracDI   AT %R3404 : DINT;   // pracovní ekvivalent proměnné delkaArchivuDI
zacatekPracC  AT %R3408 : DINT;   // pracovní ekvivalent proměnné zacatekArchivuC
delkaPracC    AT %R3412 : DINT;   // pracovní ekvivalent proměnné delkaArchivuC
zacatekPracAI AT %R3416 : DINT;   // pracovní ekvivalent proměnné zacatekArchivuAI
delkaPracAI   AT %R3420 : DINT;   // pracovní ekvivalent proměnné delkaArchivuAI
lengthDI : DINT  := sizeof(loggDI);   // délka věty
inst_RTRIGDI : R_TRIG;                // instance FB R_TRIG
positivEdgeRunDI : BOOL;              // náběžná hrana runDI
lengthC : DINT  := sizeof(loggC);     // délka věty
inst_RTRIGC : R_TRIG;                 // instance FB R_TRIG
positivEdgeRunC : BOOL;               // náběžná hrana runC
lengthAI : DINT  := sizeof(loggAI);   // délka věty
inst_RTRIGAI : R_TRIG;                // instance FB R_TRIG
positivEdgeRunAI : BOOL;              // náběžná hrana runAI
  // proměnné pro zasílání SMS
zasliSmsDI : BOOL;   //podnět pro zaslání SMS
zasliSmsC  : BOOL;
zasliSmsAI : BOOL;
mSend : BOOL;  // proměnná provede zaslání SMS
mReset : BOOL; // reset brány
mError : TGSMGateError; //Specifikace chyby při komunikaci s modemem
mState : TGSMGateStateOut; //Stav komunikace s modemem
mSignal : SINT; //Síla signálu v procentech.Hodnota -1 signalizuje neznámou úroveň signálu.
mNewMess : BOOL;  // indikace příjmu SMS
mReady : BOOL;  // signalizace, že je brána připravená na odesílání SMS
mSendPending : BOOL; //právě probíhá odesílání SMS
mSendText : SMS_STRING ;      // text SMS k odeslání
mRecvText : SMS_STRING;       // text přijaté SMS
mRecipient : NUMBER_STRING := '+420777082880';   // tel. číslo na které odesílám SMS
mSender : NUMBER_STRING;   // tel. číslo od kterého jsem přijal SMS
mCaller : NUMBER_STRING;      // tel. číslo volajícího
mCenter : NUMBER_STRING := '+420608005681'; // číslo střediska SMS Vodafone +420608005681
mPin : PIN_STRING ;  // pin sim karty
  // DataBox
velikostDBx : UINT; // velikost databoxu
offset2DI     : UDINT;  // přetypovyný ukazatel na aktuální pozici kam chci ukládat
length2DI     : USINT;  // přetypovaná délka věty
vycAddress2DI : UDINT;  // přetypovaná  počáteční adresa věty pro vyčtení
vycLength2DI  : USINT;  // Přetypovaná délka věty pro vyčtení
offset2C     : UDINT;  // přetypovyný ukazatel na aktuální pozici kam chci ukládat
length2C     : USINT;  // přetypovaná délka věty
vycAddress2C : UDINT;  // přetypovaná  počáteční adresa věty pro vyčtení
```

```
  vycLength2C  : USINT;  // Přetypovaná délka věty pro vyčtení
  offset2AI     : UDINT;  // přetypovyný ukazatel na aktuální pozici kam chci ukládat
  length2AI     : USINT;  // přetypovaná délka věty
  vycAddress2AI : UDINT;  // přetypovaná  počáteční adresa věty pro vyčtení
  vycLength2AI  : USINT;  // Přetypovaná délka věty pro vyčtení
  cesta : STRING := 'WWW/DATA/';
  ArchivAI : STRING := 'WWW/DATA/ArchivAI.CSV';
  ArchivDI : STRING := 'WWW/DATA/ArchivDI.CSV';
  ArchivC  : STRING := 'WWW/DATA/ArchivC.CSV';
END_VAR
VAR_GLOBAL CONSTANT
  NulTypDI : typDI := ( milisek := 0,  sekunda := 0,  minuta := 0,  hodina := 0, den := 0,
mesic := 0,  rok := 0,  hodnotaK1 := 0,hodnotaK2 := 0); // jeden logg (vzorek) daný strukturou
  NulTypC: typC := ( milisek := 0,  sekunda := 0,  minuta := 0,  hodina := 0, den := 0,  mesic
:= 0,  rok := 0,  hodnotaK3 := 0, hodnotaK4 := 0); // přepisovatelný archív reálných dat
  NulTypAI: typAI := ( milisek := 0,  sekunda := 0,  minuta := 0,  hodina := 0, den := 0,
mesic := 0,  rok := 0,  hodnotaK5 := 0.0, hodnotaK6 := 0.0,  hodnotaK7 := 0.0,  hodnotaK8 :=
0.0); // jeden logg (vzorek) daný strukturou
END_VAR
VAR_GLOBAL
  SendStatus : BOOL;
  g_RecvSMS : SMS_STRING;
  g_Pin : PIN_STRING :='xxxx'; // nutno zadat pin
  g_SMSCenter : NUMBER_STRING :='+420608005681';
  g_Sender : NUMBER_STRING;
  g_Recipient : NUMBER_STRING :='+420608115682';
  g_SMSText : SMS_STRING;
  g_Caller : NUMBER_STRING;
  zapisAI_na_kartu : BOOL;
  zapisC_na_kartu : BOOL;
  zapisDI_na_kartu : BOOL;
END_VAR
```

C:\MosaicApp\DiplomovaPrace\logger2\prgMain.ST

```
////////////////////////////////HLAVNÍ PROGRAM////////////////////////////////
PROGRAM prgMain
  VAR_INPUT
  END_VAR
  VAR
  iSMS : SMS_HANDLER; // instance funkčního bloku pro zasílání SMS
    ZapisAI : fbZapisNaSD;
    ZapisDI : fbZapisNaSD;
    ZapisC  : fbZapisNaSD;
  END_VAR
  VAR_OUTPUT
  END_VAR
  VAR_TEMP
    a : USINT;
    b : USINT;
    c : USINT;
    d : USINT;
    e : USINT;
    f : USINT;
 END_VAR
  ////////////////////////////// digitální kanály ////////////////////////////////
  // výběr časování pro kanály DI
  IF periodaVzDI = 0 THEN casovacDI := ms100;
  ELSIF periodaVzDI = 1 THEN casovacDI := ms500;
  ELSIF periodaVzDI = 2 THEN casovacDI := s1;
  ELSIF periodaVzDI = 3 THEN casovacDI := s10;
  ELSIF periodaVzDI = 4 THEN casovacDI := minut1;
  ELSIF periodaVzDI = 5 THEN casovacDI := minut10;
  ELSIF periodaVzDI = 6 THEN casovacDI := hod1;
  ELSIF periodaVzDI = 7 THEN casovacDI := den1;
  END_IF;
  // K1
  IF fyzCisloK1 = 0 THEN        // přiřazení fyz. kanálu k log. kanálu K1
     valK1 := r0_p3_DI.DI0;
  ELSIF fyzCisloK1 = 1 THEN
     valK1 := r0_p3_DI.DI1;
```

```
  ELSIF fyzCisloK1 = 2 THEN
     valK1 := r0_p3_DI.DI2;
  ELSIF fyzCisloK1 = 3 THEN
     valK1 := r0_p3_DI.DI3;
  END_IF;
  IF negovatK1 = TRUE THEN        // podmínka pro provedení negace
     valueK1 := not(valK1);
  ELSE valueK1 := valK1;
  END_IF;
  convertValueK1 := BOOL_TO_DINT (valueK1);  // konverze typu
  // K2
  IF fyzCisloK2 = 0 THEN        // přiřazení fyz. kanálu k log. kanálu K2
     valK2 := r0_p3_DI.DI0;
  ELSIF fyzCisloK2 = 1 THEN
     valK2 := r0_p3_DI.DI1;
  ELSIF fyzCisloK2 = 2 THEN
     valK2 := r0_p3_DI.DI2;
  ELSIF fyzCisloK2 = 3 THEN
     valK2 := r0_p3_DI.DI3;
  END_IF;
  IF negovatK2 = TRUE THEN        // podmínka pro provedení negace
     valueK2 := not(valK2);
  ELSE valueK2 := valK2;
  END_IF;
  convertValueK2 := BOOL_TO_DINT (valueK2);  // konverze typu
 // ovládání měření a archivace kanály DI
  IF runDI = TRUE THEN
     IF casovacDI = TRUE and NOT zapisDI_na_kartu THEN
        // zapsání časové značky
        realArchivDI.milisek := misek;
        realArchivDI.sekunda := sek;
        realArchivDI.minuta := minut;
        realArchivDI.hodina := hod;
        realArchivDI.den := den;
        realArchivDI.mesic := mes;
        realArchivDI.rok := rok;
       // zápis hodnoty kanál K1
        IF aktivaceK1 = TRUE THEN  // ověření aktivace kanálu
           IF aktivaceSimulK1 = FALSE THEN  // ověření aktivace simulace
              realArchivDI.hodnotaK1 := convertValueK1;
           ELSE realArchivDI.hodnotaK1 := hodnotaSimulK1;
           END_IF;
        ELSE realArchivDI.hodnotaK1 := hodnotaNeaktivDIC;
        END_IF;
        // zápis hodnoty kanál K2
        IF aktivaceK2 = TRUE THEN  // ověření aktivace kanálu
           IF aktivaceSimulK2 = FALSE THEN   // ověření aktivace simulace
              realArchivDI.hodnotaK2 := convertValueK2;
           ELSE realArchivDI.hodnotaK2 := hodnotaSimulK2;
           END_IF;
        ELSE realArchivDI.hodnotaK2 := hodnotaNeaktivDIC;
        END_IF;
        // podmínka pro archivaci a archivace
         IF cisloArchivuDI > 0 THEN
            loggDI := realArchivDI;
            offset2DI := DINT_TO_UDINT (offsetDI);
            length2DI := DINT_TO_USINT (lengthDI);
            a := WriteBlockToDBx (variable := void (loggDI), dataBoxAddress := offset2DI,
length := length2DI);
            IF offsetDI < (zacatekPracDI + delkaPracDI) THEN
               offsetDI := offsetDI + lengthDI;
               zasliSmsDI := FALSE;
            ELSE
               pocitadloDI := pocitadloDI + 1;
               zasliSmsDI := TRUE;
               zapisDI_na_kartu := true;
            END_IF;
         END_IF;
     END_IF;
  END_IF;
```

```
////////////////////////////// čítačové kanály ////////////////////////////////
   // výběr časování pro kanály C
  IF periodaVzC = 0 THEN casovacC := ms100;
  ELSIF periodaVzC = 1 THEN casovacC := ms500;
  ELSIF periodaVzC = 2 THEN casovacC := s1;
  ELSIF periodaVzC = 3 THEN casovacC := s10;
  ELSIF periodaVzC = 4 THEN casovacC := minut1;
  ELSIF periodaVzC = 5 THEN casovacC := minut10;
  ELSIF periodaVzC = 6 THEN casovacC := hod1;
  ELSIF periodaVzC = 7 THEN casovacC := den1;
  END_IF;
  // zjištění o kolik se změnila hodnota čítače v jednom cyklu programu
  citA := r0_p3_CNT_IN2.VALA;
  difA := citA - citAA;
  citAA := citA;
  convertDifA := UDINT_TO_DINT (difA);
  // zjištění o kolik se změnila hodnota čítače v jednom cyklu programu
  citB := r0_p3_CNT_IN2.VALB;
  difB := citB - citBB;
  citBB := citB;
  convertDifB := UDINT_TO_DINT (difB);
  // K3
  pvCntK3 := PVCitaceK3;
  IF pvCntK3 <> pvCntKK3  THEN //IF změna hodnoty PVCitaceK3 THEN (nastavení PV)
     valK3 := PVCitaceK3;
  END_IF;
  pvCntKK3 := PVCitaceK3;
  // K4
  pvCntK4 := PVCitaceK4;
  IF pvCntK4 <> pvCntKK4  THEN //IF změna hodnoty PVCitaceK4 THEN (nastavení PV)
     valK4 := PVCitaceK4;
  END_IF;
  pvCntKK4 := PVCitaceK4;
  // přiřazení fyz. kanálu k log. kanálu K3
  IF fyzCisloK3 = 0 THEN
     valK3 := valK3 + convertDifA;  // inktementace čítače
  ELSIF fyzCisloK3 = 1 THEN
     valK3 := valK3 + convertDifB;  // inktementace čítače
  END_IF;
  // přiřazení fyz. kanálu k log. kanálu K4
  IF fyzCisloK4 = 0 THEN
     valK4 := valK4 + convertDifA;  // inktementace čítače
  ELSIF fyzCisloK4 = 1 THEN
     valK4 := valK4 + convertDifB;  // inktementace čítače
  END_IF;
  convertValueK3 := valK3;
  convertValueK4 := valK4;
  // ovládání měření a archivace kanály C
  IF runC = TRUE THEN
     IF casovacC = TRUE and not zapisC_na_kartu THEN
        // zapsání časové značky
        realArchivC.milisek := misek;
        realArchivC.sekunda := sek;
        realArchivC.minuta := minut;
        realArchivC.hodina := hod;
        realArchivC.den := den;
        realArchivC.mesic := mes;
        realArchivC.rok := rok;
        // zápis hodnoty kanál K3
        IF aktivaceK3 = TRUE THEN  // ověření aktivace kanálu
           IF aktivaceSimulK3 = FALSE THEN  // ověření aktivace simulace
              realArchivC.hodnotaK3 := convertValueK3;
           ELSE realArchivC.hodnotaK3 := hodnotaSimulK3;
           END_IF;
        ELSE realArchivC.hodnotaK3 := hodnotaNeaktivDIC;
        END_IF;
        // zápis hodnoty kanál K4
        IF aktivaceK4 = TRUE THEN  // ověření aktivace kanálu
           IF aktivaceSimulK4 = FALSE THEN   // ověření aktivace simulace
              realArchivC.hodnotaK4 := convertValueK4;
```

```
            ELSE realArchivC.hodnotaK4 := hodnotaSimulK4;
            END_IF;
         ELSE realArchivC.hodnotaK4 := hodnotaNeaktivDIC;
         END_IF;
         // podmínka pro archivaci a archivace
          IF cisloArchivuC > 0 THEN
             loggC := realArchivC;
             offset2C := DINT_TO_UDINT (offsetC);
             length2C := DINT_TO_USINT (lengthC);
             b := WriteBlockToDBx (variable := void (loggC), dataBoxAddress := offset2C, length
:= length2C);
             IF offsetC < (zacatekPracC + delkaPracC) THEN
                offsetC := offsetC + lengthC;
                zasliSmsC := FALSE;
             ELSE
                pocitadloC := pocitadloC + 1;
                zasliSmsC := TRUE;
                zapisC_na_kartu := true;
             END_IF;
          END_IF;
      END_IF;
   END_IF;
   ////////////////////////////// analogové kanály ////////////////////////////
   // výběr časování pro kanály AI
   IF periodaVzAI = 0 THEN casovacAI := ms100;
   ELSIF periodaVzAI = 1 THEN casovacAI := ms500;
   ELSIF periodaVzAI = 2 THEN casovacAI := s1;
   ELSIF periodaVzAI = 3 THEN casovacAI := s10;
   ELSIF periodaVzAI = 4 THEN casovacAI := minut1;
   ELSIF periodaVzAI = 5 THEN casovacAI := minut10;
   ELSIF periodaVzAI = 6 THEN casovacAI := hod1;
   ELSIF periodaVzAI = 7 THEN casovacAI := den1;
   END_IF;
   // K5
   IF fyzCisloK5 = 0 THEN        // přiřazení fyz. kanálu k log. kanálu K5
      valK5 := r0_p3_AI0.ENG;
   ELSIF fyzCisloK5 = 1 THEN
      valK5 := r0_p3_AI1.ENG;
   ELSIF fyzCisloK5 = 2 THEN
      valK5 := r0_p3_AI2.ENG;
   ELSIF fyzCisloK5 = 3 THEN
      valK5 := r0_p3_AI3.ENG;
   END_IF;
   // přepočet vstup. napěťového signálu na rozsah daný uživatelem
   koefK5 := (techRozsahHMK5 - techRozsahDMK5)/(elRozsahHMK5 - elRozsahDMK5);
   valueK5 := (koefK5 * valK5) + techRozsahDMK5;
   // K6
   IF fyzCisloK6 = 0 THEN        // přiřazení fyz. kanálu k log. kanálu K6
      valK6 := r0_p3_AI0.ENG;
   ELSIF fyzCisloK6 = 1 THEN
      valK6 := r0_p3_AI1.ENG;
   ELSIF fyzCisloK6 = 2 THEN
      valK6 := r0_p3_AI2.ENG;
   ELSIF fyzCisloK6 = 3 THEN
      valK6 := r0_p3_AI3.ENG;
   END_IF;
   koefK6 := (techRozsahHMK6 - techRozsahDMK6)/(elRozsahHMK6 - elRozsahDMK6);
   valueK6 := (koefK6 * valK6) + techRozsahDMK6;
   // K7
   IF fyzCisloK7 = 0 THEN        // přiřazení fyz. kanálu k log. kanálu K7
      valK7 := r0_p3_AI0.ENG;
   ELSIF fyzCisloK7 = 1 THEN
      valK7 := r0_p3_AI1.ENG;
   ELSIF fyzCisloK7 = 2 THEN
      valK7 := r0_p3_AI2.ENG;
   ELSIF fyzCisloK7 = 3 THEN
      valK7 := r0_p3_AI3.ENG;
   END_IF;
   koefK7 := (techRozsahHMK7 - techRozsahDMK7)/(elRozsahHMK7 - elRozsahDMK7);
   valueK7 := (koefK7 * valK7) + techRozsahDMK7;
```

```
   // K8
  IF fyzCisloK8 = 0 THEN        // přiřazení fyz. kanálu k log. kanálu K8
     valK8 := r0_p3_AI0.ENG;
  ELSIF fyzCisloK8 = 1 THEN
     valK8 := r0_p3_AI1.ENG;
  ELSIF fyzCisloK8 = 2 THEN
     valK8 := r0_p3_AI2.ENG;
  ELSIF fyzCisloK8 = 3 THEN
     valK8 := r0_p3_AI3.ENG;
  END_IF;
  koefK8 := (techRozsahHMK8 - techRozsahDMK8)/(elRozsahHMK8 - elRozsahDMK8);
  valueK8 := (koefK8 * valK8) + techRozsahDMK8;
  // ovládání měření a archivace kanály AI
  IF runAI = TRUE THEN
     IF casovacAI and not zapisAI_na_kartu THEN
        // zapsání časové značky
        realArchivAI.milisek := misek;
        realArchivAI.sekunda := sek;
        realArchivAI.minuta := minut;
        realArchivAI.hodina := hod;
        realArchivAI.den := den;
        realArchivAI.mesic := mes;
        realArchivAI.rok := rok;
        // zápis hodnoty kanál K5
        IF aktivaceK5 = TRUE THEN  // ověření aktivace kanálu
           IF aktivaceSimulK5 = FALSE THEN  // ověření aktivace simulace
              realArchivAI.hodnotaK5 := valueK5;
           ELSE realArchivAI.hodnotaK5 := hodnotaSimulK5;
           END_IF;
        ELSE realArchivAI.hodnotaK5 := hodnotaNeaktivAI;
        END_IF;
        // zápis hodnoty kanál K6
        IF aktivaceK6 = TRUE THEN  // ověření aktivace kanálu
           IF aktivaceSimulK6 = FALSE THEN  // ověření aktivace simulace
              realArchivAI.hodnotaK6 := valueK6;
           ELSE realArchivAI.hodnotaK6 := hodnotaSimulK6;
           END_IF;
        ELSE realArchivAI.hodnotaK6 := hodnotaNeaktivAI;
        END_IF;
           // zápis hodnoty kanál K7
        IF aktivaceK7 = TRUE THEN  // ověření aktivace kanálu
           IF aktivaceSimulK7 = FALSE THEN  // ověření aktivace simulace
              realArchivAI.hodnotaK7 := valueK7;
           ELSE realArchivAI.hodnotaK7 := hodnotaSimulK7;
           END_IF;
        ELSE realArchivAI.hodnotaK7 := hodnotaNeaktivAI;
        END_IF;
          // zápis hodnoty kanál K8
        IF aktivaceK8 = TRUE THEN  // ověření aktivace kanálu
           IF aktivaceSimulK8 = FALSE THEN  // ověření aktivace simulace
              realArchivAI.hodnotaK8 := valueK8;
           ELSE realArchivAI.hodnotaK8 := hodnotaSimulK8;
           END_IF;
        ELSE realArchivAI.hodnotaK8 := hodnotaNeaktivAI;
        END_IF;
        // podmínka pro archivaci a archivace
         IF cisloArchivuAI > 0 THEN
            loggAI := realArchivAI;
            offset2AI := DINT_TO_UDINT (offsetAI);
            length2AI := DINT_TO_USINT (lengthAI);
            c := WriteBlockToDBx (variable := void (loggAI), dataBoxAddress := offset2AI,
length := length2AI);
            offsetAI := offsetAI + lengthAI;
            IF offsetAI < (zacatekPracAI + delkaPracAI) THEN
               zasliSmsAI := FALSE;
            ELSE
               pocitadloAI := pocitadloAI + 1;
               zasliSmsAI := TRUE;
               zapisAI_na_kartu := true;
            END_IF;
```

```
        END_IF;
      END_IF;
  END_IF;
  // Zapis na kartu
  ZapisDI(start := zapisDI_na_kartu, zdroj :=zacatekArchivuDI , delkaArchivu :=offsetDI-
zacatekPracDI , JmenoArchivu :=ArchivDI, datatype := 0 );
  IF ZapisDI.hotovo OR ZapisDI.chyba THEN
     zapisDI_na_kartu := false;
     offsetDI := zacatekPracDI;
  END_IF;
  ZapisC(start := zapisC_na_kartu, zdroj :=zacatekArchivuC , delkaArchivu :=offsetC-
zacatekPracC , JmenoArchivu :=ArchivC, datatype := 1 );
  IF ZapisC.hotovo OR ZapisC.chyba THEN
     zapisC_na_kartu := false;
     offsetC := zacatekPracC;
  END_IF;
  ZapisAI(start := zapisAI_na_kartu, zdroj :=zacatekArchivuAI , delkaArchivu :=offsetAI-
zacatekPracAI , JmenoArchivu :=ArchivAI, datatype := 2 );
  IF ZapisAI.hotovo OR ZapisAI.chyba THEN
     zapisAI_na_kartu := false;
     offsetAI := zacatekPracAI;
  END_IF;
  ////////////////////////////další funkce/////////////////////////////////////
    // vyčítání archívu DI
  IF vycControlDI = 1 THEN
     vycAddress2DI := DINT_TO_UDINT (vycAddressDI);
     vycLength2DI := INT_TO_USINT (vycLengthDI);
     d := ReadBlockFromDBx(dataBoxAddress := vycAddress2DI, length := vycLength2DI, variable
:= void(vycBufferDI));
     vycControlDI := 0;
  END_IF;
  IF promPocitadloDI < pocitadloDI  and promoticPointerDI = (offsetDI + lengthDI) THEN
     IF offsetDI < (zacatekPracDI + delkaPracDI) THEN
        promoticPointerDI := offsetDI + lengthDI + lengthDI;
     ELSE
       promoticPointerDI := zacatekPracDI + lengthDI + lengthDI;
     END_IF;
     straceneZaznamyDI := straceneZaznamyDI + 1;
  ELSIF promPocitadloDI < pocitadloDI  and promoticPointerDI = 0 THEN
       promoticPointerDI := zacatekPracDI + lengthDI + lengthDI;
       straceneZaznamyDI := straceneZaznamyDI + 1;
  END_IF;
   // vyčítání archívu C
  IF vycControlC = 1 THEN
     vycAddress2C := DINT_TO_UDINT (vycAddressC);
     vycLength2C := INT_TO_USINT (vycLengthC);
     e := ReadBlockFromDBx(dataBoxAddress := vycAddress2C, length := vycLength2C, variable :=
void(vycBufferC));
     vycControlC := 0;
  END_IF;
  IF promPocitadloC < pocitadloC  and promoticPointerC = (offsetC + lengthC) THEN
     IF offsetC < (zacatekPracC + delkaPracC) THEN
        promoticPointerC := offsetC + lengthC + lengthC;
     ELSE
       promoticPointerC := zacatekPracC + lengthC + lengthC;
     END_IF;
     straceneZaznamyC := straceneZaznamyC + 1;
  ELSIF promPocitadloC < pocitadloC  and promoticPointerC = 0 THEN
       promoticPointerC := zacatekPracC + lengthC + lengthC;
       straceneZaznamyC := straceneZaznamyC + 1;
  END_IF;
     // vyčítání archívu AI
  IF vycControlAI = 1 THEN
     vycAddress2AI := DINT_TO_UDINT (vycAddressAI);
     vycLength2AI := INT_TO_USINT (vycLengthAI);
     f := ReadBlockFromDBx(dataBoxAddress := vycAddress2AI, length := vycLength2AI, variable
:= void(vycBufferAI));
     vycControlAI := 0;
  END_IF;
  IF promPocitadloAI < pocitadloAI  and promoticPointerAI = (offsetAI + lengthAI) THEN
```

```
    IF offsetAI < (zacatekPracAI + delkaPracAI) THEN
       promoticPointerAI := offsetAI + lengthAI + lengthAI;
    ELSE
      promoticPointerAI := zacatekPracAI + lengthAI + lengthAI;
    END_IF;
    straceneZaznamyAI := straceneZaznamyAI + 1;
  ELSIF promPocitadloAI < pocitadloAI  and promoticPointerAI = 0 THEN
       promoticPointerAI := zacatekPracAI + lengthAI + lengthAI;
       straceneZaznamyAI := straceneZaznamyAI + 1;
  END_IF;
  //zápis hodnot některých proměnných pro komunikaci
  delkaVetyDI := lengthDI;   //délka věty v B
  delkaVetyC := lengthC;
  delkaVetyAI := lengthAI;
  ukazatelDI := offsetDI; // počet B od R0
  ukazatelC := offsetC;
  ukazatelAI := offsetAI;
  velikostDBx := SizeOfDataBox();  // zjištění velikosti DataBoxu
END_PROGRAM
```

C:\MosaicApp\DiplomovaPrace\logger2\prgCasovy.ST

```
PROGRAM prgCasovy
  VAR_INPUT
  END_VAR
  VAR
  END_VAR
  VAR_OUTPUT
  END_VAR
  VAR_IN_OUT
  END_VAR
  VAR_EXTERNAL
  END_VAR
  VAR_TEMP
  END_VAR
  IF restartDataloggeru = 1 THEN
    // K1,DI datová struktura
     fyzCisloK1       :=0;             //číslo fyzického kanálu
     logCisloK1       :=1;             //číslo logického kanálu
     nazevK1          :='digitalni 1'; //název kanálu
     jednotkaK1       :='digit';       //fyzikální jednotka
     aktivaceK1       :=TRUE;          //aktivace měření kanálu
     typVstupK1       :=1;             //výběr typu vstupu
     negovatK1        :=FALSE;         //negovat vstupní signál
     PVCitaceK1       :=0;             //přednastavená hodnota čítače
     aktivaceSimulK1  :=FALSE;         //povolení ruční simulované hodnoty
     hodnotaSimulK1   :=0;             //zadaní ruční simulované hodnoty
     cisloArchivuK1   :=0;             //přiřazení archivu k tomuto kanalu
     // K2,DI datová struktura
     fyzCisloK2       :=1;             //číslo fyzického kanálu
     logCisloK2       :=2;             //číslo logického kanálu
     nazevK2          :='digitalni 2'; //název kanálu
     jednotkaK2       :='digit';       //fyzikální jednotka
     aktivaceK2       :=TRUE;          //aktivace měření kanálu
     typVstupK2       :=1;             //výběr typu vstupu
     negovatK2        :=FALSE;         //negovat vstupní signál
     PVCitaceK2       :=0;             //přednastavená hodnota čítače
     aktivaceSimulK2  :=FALSE;         //povolení ruční simulované hodnoty
     hodnotaSimulK2   :=0;             //zadaní ruční simulované hodnoty
     cisloArchivuK2   :=0;             //přiřazení archivu k tomuto kanalu
     // K3,DI datová struktura
     fyzCisloK3       :=0;          //číslo fyzického kanálu
     logCisloK3       :=3;          //číslo logického kanálu
     nazevK3          := 'čítač1';  //název kanálu
     jednotkaK3       := 'impuls';  //fyzikální jednotka
     aktivaceK3       := TRUE;      //aktivace měření kanálu
     typVstupK3       :=3;          //výběr typu vstupu
     negovatK3        :=FALSE;      //negovat vstupní signál
     PVCitaceK3       :=0;          //přednastavená hodnota čítače
     aktivaceSimulK3  :=FALSE;      //povolení ruční simulované hodnoty
     hodnotaSimulK3   :=0;          //zadaní ruční simulované hodnoty
```

```
cisloArchivuK3   :=0;          //přiřazení archivu k tomuto kanalu
// K4,DI datová struktura
fyzCisloK4       :=1;         //číslo fyzického kanálu
logCisloK4       :=4;         //číslo logického kanálu
nazevK4          :='čítač 2'; //název kanálu
jednotkaK4       :='impuls';  //fyzikální jednotka
aktivaceK4       :=TRUE;      //aktivace měření kanálu
typVstupK4       :=3;         //výběr typu vstupu
negovatK4        :=FALSE;     //negovat vstupní signál
PVCitaceK4       :=0;         //přednastavená hodnota čítače
aktivaceSimulK4  :=FALSE;     //povolení ruční simulované hodnoty
hodnotaSimulK4   :=0;         //zadaní ruční simulované hodnoty
cisloArchivuK4   :=0;         //přiřazení archivu k tomuto kanalu
// K5,AI datová struktura
fyzCisloK5       :=0;             //číslo fyzického kanálu
logCisloK5       :=5;             //číslo logického kanálu
nazevK5          :='analogový 1'; //název kanálu
jednotkaK5       :='[V]';         //fyzikální jednotka
aktivaceK5       :=TRUE;          //aktivace měření kanálu
typDatK5         :=3;             //výběr reprezentace dat ADC
elRozsahHMK5     :=10.0;          //zadání horní meze rozsahu el. signálu (výstup z ADC)
elRozsahDMK5     :=0.0;           //zadání dolní meze rozsahu el. signálu (výstup z ADC)
techRozsahHMK5   :=10.0;          //zadání horní meze rozsahu tecnologického signálu
techRozsahDMK5   :=0.0;           //zadání dolní meze rozsahu tecnologického signálu
aktivaceSimulK5  :=FALSE;         //povolení ruční simulované hodnoty
hodnotaSimulK5   :=0.0;             //zadaní ruční simulované hodnoty
cisloArchivuK5   :=0;             //přiřazení archivu k tomuto kanalu
// K6,AI datová struktura
fyzCisloK6       :=1;             //číslo fyzického kanálu
logCisloK6       :=6;             //číslo logického kanálu
nazevK6          :='analogový 2'; //název kanálu
jednotkaK6       :='[V]';         //fyzikální jednotka
aktivaceK6       :=TRUE;          //aktivace měření kanálu
typDatK6         :=3;             //výběr reprezentace dat ADC
elRozsahHMK6     :=10.0;          //zadání horní meze rozsahu el. signálu (výstup z ADC)
elRozsahDMK6     :=0.0;           //zadání dolní meze rozsahu el. signálu (výstup z ADC)
techRozsahHMK6   :=10.0;          //zadání horní meze rozsahu tecnologického signálu
techRozsahDMK6   :=0.0;           //zadání dolní meze rozsahu tecnologického signálu
aktivaceSimulK6  :=FALSE;         //povolení ruční simulované hodnoty
hodnotaSimulK6   :=0.0;             //zadaní ruční simulované hodnoty
cisloArchivuK6   :=0;             //přiřazení archivu k tomuto kanalu
// K7,AI datová struktura
fyzCisloK7       :=2;             //číslo fyzického kanálu
logCisloK7       :=7;             //číslo logického kanálu
nazevK7          :='analogový 3'; //název kanálu
jednotkaK7       :='[V]';         //fyzikální jednotka
aktivaceK7       :=TRUE;          //aktivace měření kanálu
typDatK7         :=3;             //výběr reprezentace dat ADC
elRozsahHMK7     :=10.0;          //zadání horní meze rozsahu el. signálu (výstup z ADC)
elRozsahDMK7     :=0.0;           //zadání dolní meze rozsahu el. signálu (výstup z ADC)
techRozsahHMK7   :=10.0;          //zadání horní meze rozsahu tecnologického signálu
techRozsahDMK7   :=0.0;           //zadání dolní meze rozsahu tecnologického signálu
aktivaceSimulK7  :=FALSE;         //povolení ruční simulované hodnoty
hodnotaSimulK7   :=0.0;             //zadaní ruční simulované hodnoty
cisloArchivuK7   :=0;             //přiřazení archivu k tomuto kanalu
// K8,AI datová struktura
fyzCisloK8       :=3;             //číslo fyzického kanálu
logCisloK8       :=8;             //číslo logického kanálu
nazevK8          :='analogový 4'; //název kanálu
jednotkaK8       :='[V]';         //fyzikální jednotka
aktivaceK8       :=TRUE;          //aktivace měření kanálu
typDatK8         :=3;             //výběr reprezentace dat ADC
elRozsahHMK8     :=10.0;          //zadání horní meze rozsahu el. signálu (výstup z ADC)
elRozsahDMK8     :=0.0;           //zadání dolní meze rozsahu el. signálu (výstup z ADC)
techRozsahHMK8   :=10.0;          //zadání horní meze rozsahu tecnologického signálu
techRozsahDMK8   :=0.0;           //zadání dolní meze rozsahu tecnologického signálu
aktivaceSimulK8  :=FALSE;         //povolení ruční simulované hodnoty
hodnotaSimulK8   :=0.0;             //zadaní ruční simulované hodnoty
cisloArchivuK8   :=0;             //přiřazení archivu k tomuto kanalu
//globální nastavovací prvky
```

```
aktivaceSMS    :=FALSE;           //povolení zasílíání SMS
telCisloSMS    :='+420XXXXXXXXX'; // tel. číslo pro příjem SMS
//ovládání start stop měření a perioda vzorkování pro DI (K1, K2)
periodaVzDI         :=0;     // perioda vzorkování
rucniStartStopDI    :=FALSE; //ruční nadřazený start/stop měření
//ovládání start stop měření a perioda vzorkování pro čítače C (K3, K4)
periodaVzC         := 0;      // perioda vzorkování
rucniStartStopC    :=FALSE;  //ruční nadřazený start/stop měření
//ovládání start stop měření a perioda vzorkování pro AI (K5, K6, K7, K8)
periodaVzAI         :=0;       // perioda vzorkování
rucniStartStopAI    :=FALSE;  //ruční nadřazený start/stop měření
//reálné archívy
realArchivDI.milisek   := 0;
realArchivDI.sekunda   := 0;
realArchivDI.minuta    := 0;
realArchivDI.hodina    := 0;
realArchivDI.den       := 0;
realArchivDI.mesic     := 0;
realArchivDI.rok       := 0;
realArchivDI.hodnotaK1 := 0;
realArchivDI.hodnotaK2 := 0;
realArchivC.milisek   := 0;
realArchivC.sekunda   := 0;
realArchivC.minuta    := 0;
realArchivC.hodina    := 0;
realArchivC.den       := 0;
realArchivC.mesic     := 0;
realArchivC.rok       := 0;
realArchivC.hodnotaK3 := 0;
realArchivC.hodnotaK4 := 0;
realArchivAI.milisek   := 0;
realArchivAI.sekunda   := 0;
realArchivAI.minuta    := 0;
realArchivAI.hodina    := 0;
realArchivAI.den       := 0;
realArchivAI.mesic     := 0;
realArchivAI.rok       := 0;
realArchivAI.hodnotaK5 := 0.0;
realArchivAI.hodnotaK6 := 0.0;
realArchivAI.hodnotaK7 := 0.0;
realArchivAI.hodnotaK8 := 0.0;
//archív DI
cisloArchivuDI    :=1;       //identifikační číslo archívu
zacatekArchivuDI  :=0;       //počáteční adresa archívu v paměti
delkaArchivuDI    :=1000;  //délka archívu, puvodne 150000
delkaVetyDI       :=0;       //délka jednoho záznamu v archívu
ukazatelDI        :=0;       //ukazatel na aktuální pozici v archívu
loggDI            := NulTypDI;   // jeden logg (vzorek) daný strukturou
realArchivDI      := NulTypDI;   // přepisovatelný archív reálných dat tvořen strukturou
//archív čítače C
cisloArchivuC   :=1;      //identifikační číslo archívu
zacatekArchivuC :=150000; //počáteční adresa archívu v paměti
delkaArchivuC   :=1000; //délka archívu
delkaVetyC      :=0;      //délka jednoho záznamu v archívu
ukazatelC       :=0;      //ukazatel na aktuální pozici v archívu
loggC           := NulTypC;   // jeden logg (vzorek) daný strukturou
realArchivC     := NulTypC;   // přepisovatelný archív reálných dat tvořen strukturou
//archív AI
cisloArchivuAI   :=1;      //identifikační číslo archívu
zacatekArchivuAI :=300000; //počáteční adresa archívu v paměti
delkaArchivuAI   :=1000; //délka archívu
delkaVetyAI      :=0;      //délka jednoho záznamu v archívu
ukazatelAI       :=0;      //ukazatel na aktuální pozici v archívu
loggAI           := NulTypAI;   // jeden logg (vzorek) daný strukturou
realArchivAI     := NulTypAI;   // přepisovatelný archív reálných dat tvořen strukturou
// stahování archívů DI
promoticPointerDI :=0;
promPocitadloDI :=0;
straceneZaznamyDI :=0; // počet stracených vět při vyčítání dat
// stahování archívů C
```

```
      promoticPointerC :=0;     promPocitadloC :=0;
      straceneZaznamyC :=0;
      promoticPointerAI :=0;
      promPocitadloAI  :=0;     straceneZaznamyAI :=0; // počet stracených vět při vyčítání dat
      // vnitřní proměnné
      valK3 :=0; // obraz přiřezeného čítače
      valK4 :=0; // obraz přiřazeného čítače
      citA   := r0_p3_CNT_IN2.VALA;  //hodnota čítače A (r0_p3_CNT_IN2.VALA)
      citAA  := r0_p3_CNT_IN2.VALA;  //hodnota čítače A v předchozím cyklu programu
      difA   :=0;                    //změna hodnoty čítače v cyklu programu
      convertDifA :=0;               //convetr předchozí proměnné
      citB   := r0_p3_CNT_IN2.VALB;  //hodnota čítače B (r0_p3_CNT_IN2.VALB)
      citBB  := r0_p3_CNT_IN2.VALB;  //hodnota čítače B v předchozím cyklu programu
      difB   :=0;                    //změna hodnoty čítače v cyklu programu
      convertDifB :=0;               //convetr předchozí proměnné
      pocitadloDI :=0;  //počítadlo počtu přepisů archívu DI
      pocitadloC  :=0;  //počítadlo počtu přepisů archívu C
      pocitadloAI :=0;  //počítadlo počtu přepisů archívu AI
      offsetDI := zacatekArchivuDI;  // ukazatel na aktuální pozici kam chci ukládat
      offsetC  := zacatekArchivuC;   // ukazatel na aktuální pozici kam chci ukládat
      offsetAI := zacatekArchivuAI;  // ukazatel na aktuální pozici kam chci ukládat
      // restart do nuly
      restartDataloggeru := 0;
   END_IF;
   ////////////////////////////////   DI  ////////////////////////////////
   IF rucniStartStopDI = TRUE THEN
      runDI := TRUE;
   ELSE runDI := FALSE;
   END_IF;
    // zjištění změny začátku nebo délky archívu
   zacatek1DI := zacatekArchivuDI;
   IF zacatek1DI <> zacatek2DI  THEN
      zmenaDI := TRUE;
   END_IF;
   zacatek2DI := zacatekArchivuDI;
   delka1DI := delkaArchivuDI;
   IF delka1DI <> delka2DI   THEN
      zmenaDI := TRUE;
   END_IF;
   delka2DI  := delkaArchivuDI;
// nastavení počáteční adresy archívu při přechodu do run
   inst_RTRIGDI( CLK := runDI, Q => positivEdgeRunDI);
   IF positivEdgeRunDI = TRUE AND zmenaDI = TRUE THEN
      offsetDI := zacatekArchivuDI;
      zacatekPracDI := zacatekArchivuDI;
      delkaPracDI := delkaArchivuDI - lengthDI;
      pocitadloDI := 0;
      promoticPointerDI :=0;
      promPocitadloDI :=0;
      straceneZaznamyDI :=0;
      zmenaDI := FALSE;
   END_IF;
   ////////////////////////////////   C   ////////////////////////////////
   IF rucniStartStopC = TRUE THEN
      runC := TRUE;
   ELSE runC := FALSE;
   END_IF;
   // zjištění změny začátku nebo délky archívu
   zacatek1C := zacatekArchivuC;
   IF zacatek1C <> zacatek2C  THEN
      zmenaC := TRUE;
   END_IF;
   zacatek2C := zacatekArchivuC;
   delka1C := delkaArchivuC;
   IF delka1C <> delka2C   THEN
      zmenaC := TRUE;
   END_IF;
   delka2C  := delkaArchivuC;
    // nastavení počáteční adresy archívu při přechodu do run
   inst_RTRIGC( CLK := runC, Q => positivEdgeRunC);
```

```
  IF positivEdgeRunC = TRUE AND zmenaC = TRUE THEN
     offsetC := zacatekArchivuC;
     pocitadloC := 0;
     zacatekPracC := zacatekArchivuC;
     delkaPracC := delkaArchivuC - lengthC;
     promoticPointerC :=0;
     promPocitadloC :=0;
     straceneZaznamyC :=0;
     zmenaC := FALSE;
  END_IF;
  // ovladani citacu  kanal K3 a K4
  IF enableCnt = TRUE THEN              // povoleni citacu nastaveno napevno
     r0_p3_CNT_OUT2.CCNT.0 := 1;
     r0_p3_CNT_OUT2.CCNT.8 := 1;
  ELSE
     r0_p3_CNT_OUT2.CCNT.0 := 0;
     r0_p3_CNT_OUT2.CCNT.8 := 0;
  END_IF;
  ////////////////////////////////   AI   ///////////////////////////////////
  IF rucniStartStopAI = TRUE THEN
     runAI := TRUE;
  ELSE runAI := FALSE;
  END_IF;
  // zjištění změny začátku nebo délky archívu
  zacatek1AI := zacatekArchivuAI;
  IF zacatek1AI <> zacatek2AI  THEN
     zmenaAI := TRUE;
  END_IF;
  zacatek2AI := zacatekArchivuAI;
  delka1AI := delkaArchivuAI;
  IF delka1AI <> delka2AI   THEN
     zmenaAI := TRUE;
  END_IF;
  delka2AI  := delkaArchivuAI;
  inst_RTRIGAI( CLK := runAI, Q => positivEdgeRunAI);
  IF positivEdgeRunAI = TRUE AND zmenaAI = TRUE THEN
     offsetAI := zacatekArchivuAI;
     pocitadloAI := 0;
     zacatekPracAI := zacatekArchivuAI;
     delkaPracAI := delkaArchivuAI - lengthAI;
     promoticPointerAI :=0;
     promPocitadloAI :=0;
     straceneZaznamyAI :=0;
     zmenaAI := FALSE;
  END_IF;
 END_PROGRAM
```

C:\MosaicApp\DiplomovaPrace\logger2\TO_UPPER.ST

```
FUNCTION TO_UPPER : USINT
  VAR_TEMP
    p : PTR_TO BYTE;
  END_VAR
  VAR_IN_OUT
    IN : SMS_STRING;
  END_VAR
 p := ADR(IN);
  WHILE p^ <> 0 DO              //till the end of string
    IF (p^ > 16#60) AND (p^ < 16#7B) THEN          //if character is lower case
      p^ := p^ & 2#11011111;
    END_IF;
    p := p+1;
  END_WHILE;
   TO_UPPER := UDINT_TO_USINT(PTR_TO_UDINT(p) - PTR_TO_UDINT(ADR(IN)));
END_FUNCTION
```

C:\MosaicApp\DiplomovaPrace\logger2\CompareSMS.ST

```
FUNCTION CompareSMS : BOOL
  VAR_INPUT
    New : BOOL;          //new message to compare
  END_VAR
```

```
  VAR
  END_VAR
  VAR_IN_OUT
    In  : SMS_STRING;    //message to compare
  END_VAR
  VAR_TEMP
    lng : USINT;
  END_VAR
   CompareSMS := false;
   IF New THEN //New THEN                  //if there are new message to compare
    lng := TO_UPPER(In);       //convert received message to upper case 'abC' => 'ABC'
    IF In = 'POSLAT' THEN
      CompareSMS := true;
    END_IF;
    IF In = 'STARTDI'  THEN
     rucniStartStopDI := TRUE;
    ELSIF In = 'STARTC' THEN
     rucniStartStopC := TRUE;
    ELSIF In = 'STARTAI' THEN
     rucniStartStopAI := TRUE;
    END_IF;
    IF In = 'STOPDI'  THEN
     rucniStartStopDI := FALSE;
    ELSIF In = 'STOPC' THEN
     rucniStartStopC := FALSE;
    ELSIF In = 'STOPAI' THEN
     rucniStartStopAI := FALSE;
    END_IF;
    IF In = 'ULOZ DI'  THEN
     zapisDI_na_kartu := true;
    ELSIF In = 'ULOZ C' THEN
     zapisC_na_kartu := true;
    ELSIF In = 'ULOZ AI' THEN
     zapisAI_na_kartu := true;
    END_IF;
   END_IF;
 END_FUNCTION
```

C:\MosaicApp\DiplomovaPrace\logger2\fbMakeSMS.ST

```
FUNCTION_BLOCK fbMakeSMS
  VAR_INPUT
  END_VAR
  VAR
    Comma : STRING[1] :=';';
  END_VAR
  VAR_IN_OUT
    SMSText : SMS_STRING;       // string variable must be IN_OUT type
  END_VAR
  VAR_OUTPUT
  END_VAR
  VAR_TEMP
  END_VAR
  SMSText := DT_TO_STRINGF(in := GetDateTime(), format := '%TYYYY-MM-DD-hh:mm:ss') +
             Comma + BOOL_TO_STRING(realArchivDI.hodnotaK1) +
             Comma + BOOL_TO_STRING(realArchivDI.hodnotaK2) +
             Comma + DINT_TO_STRING(realArchivC.hodnotaK3) +
             Comma + DINT_TO_STRING(realArchivC.hodnotaK4) +
             Comma + REAL_TO_STRINGF(in := realArchivAI.hodnotaK5, format := '%5.1f') +
             Comma + REAL_TO_STRINGF(in := realArchivAI.hodnotaK6, format := '%5.1f') +
             Comma + REAL_TO_STRINGF(in := realArchivAI.hodnotaK7, format := '%5.1f') +
             Comma + REAL_TO_STRINGF(in := realArchivAI.hodnotaK8, format := '%5.1f');
END_FUNCTION_BLOCK
```

C:\MosaicApp\DiplomovaPrace\logger2\WriteDbxToFileText.ST

```
FUNCTION_BLOCK WriteDbxToFileText
  VAR_IN_OUT
    fileName         : string [80];  (* file name (including path)*)
  END_VAR
  VAR_INPUT
    exec             : BOOL R_EDGE; (* request (rising edge)*)
```

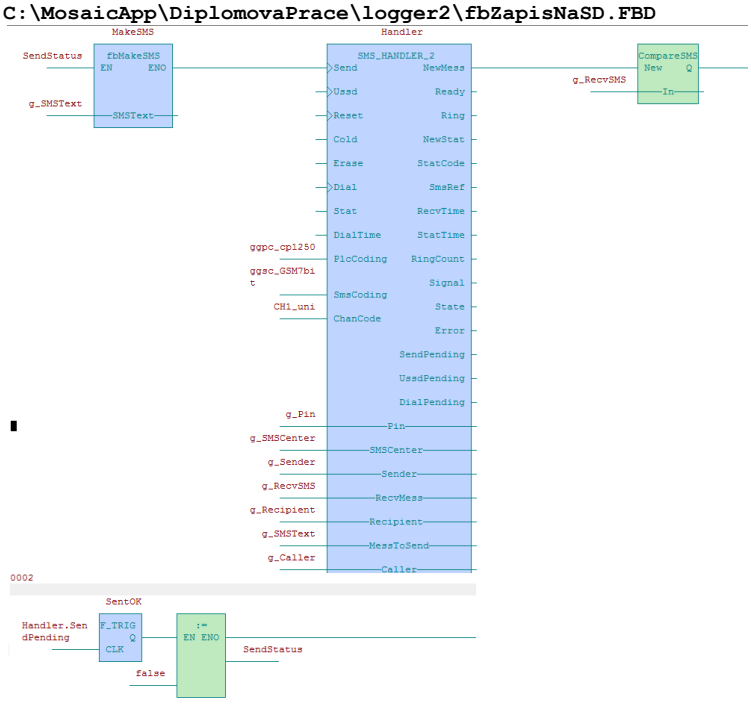
```
    seek              : udint;   (* data offset in file*)
    srcAdr            : udint;   (* databox address*)
    size              : udint;   (* data length (number of bytes)*)
    datatype          : usint;   (* type of data 0 - typDI, 1 - typC, 2 - typAI *)
  END_VAR
  VAR_OUTPUT
    done              : bool;    (* action is done*)
    busy              : bool;    (* action in progress*)
    err               : bool;    (* error flag*)
    errID             : udint;   (* error number (0 = no error)*)
    actSize           : udint;   (* number of bytes really written*)
  END_VAR
  VAR
    n                 : udint;
    n1                : udint;
    na                : udint;
    i                 : uint;
    j                 : uint;
    s                 : uint;
    sizeofstruct      : udint;
    savedsize         : udint;
    bufferbin         : array [0..255] of byte;
    bufferTXT         : array [0..1023] of byte;
    sline             : STRING;
    ptrDI             : PTR_TO typDI;
    ptrC              : PTR_TO typC;
    ptrAI             : PTR_TO typAI;
    iWriteToFile      : WriteToFile;
    savetofile        : BOOL;
  END_VAR
   done := false;
    IF exec THEN
     CASE datatype OF
       0 : sizeofstruct := SIZEOF(typDI);
       1 : sizeofstruct := SIZEOF(typC);
       2 : sizeofstruct := SIZEOF(typAI);
     END_CASE;
     savedsize := 0;
     n1 := 256  / sizeofstruct;
     busy := true;
   END_IF;
   err := false;
iWriteToFile(fileName := fileName, srcVar := void(bufferTXT), exec := savetofile, seek :=seek,
               size := UINT_TO_UDINT(j), err => err, errID => errID, actSize => actSize);
   savetofile := false;
   IF busy THEN
     IF err THEN
       busy := false;
       return;
     END_IF;
   IF NOT iWriteToFile.done AND NOT iWriteToFile.busy THEN //pokud neukladam pripravit zaznam
       n  := (size-savedsize) / sizeofstruct;
       na := min(n1, n);
       savedsize := savedsize + USINT_TO_UDINT(ReadBlockFromDBx(variable := void(bufferbin),
dataBoxAddress := srcAdr + savedsize, length := UDINT_TO_USINT(na * sizeofstruct)));
     IF na > 0 THEN
         j := 0;
         FOR i := 0 TO UDINT_TO_UINT(na-1) DO
           ptrDI := ADR(bufferbin) + UINT_TO_UDINT(i)*sizeofstruct;
           ptrC  := ADR(bufferbin) + UINT_TO_UDINT(i)*sizeofstruct;
           ptrAI := ADR(bufferbin) + UINT_TO_UDINT(i)*sizeofstruct;
           sline :=   USINT_TO_STRINGF(in := ptrDI^.rok, format := '20%02d-')+
USINT_TO_STRINGF(in := ptrDI^.mesic, format := '%02d-') +
                      USINT_TO_STRINGF(in := ptrDI^.den, format := '%02d-') +
USINT_TO_STRINGF(in := ptrDI^.hodina, format := '%02d:') +
                      USINT_TO_STRINGF(in := ptrDI^.minuta, format := '%02d:')  +
USINT_TO_STRINGF(in := ptrDI^.sekunda, format := '%02d:') +
                      USINT_TO_STRINGF(in := ptrDI^.milisek, format := '%02d;');
          CASE datatype OF
             0 : sline := sline + DINT_TO_STRING(ptrDI^.hodnotaK1) + ';' +
```

```
DINT_TO_STRING(ptrDI^.hodnotaK2) + '$r$n';
              1 : sline := sline + DINT_TO_STRING(ptrC^.hodnotaK3) + ';' +
DINT_TO_STRING(ptrC^.hodnotaK4) + '$r$n';
              2 : sline := sline + REAL_TO_STRING(ptrAI^.hodnotaK5) + ';' +
REAL_TO_STRING(ptrAI^.hodnotaK6) + ';' +
                                  REAL_TO_STRING(ptrAI^.hodnotaK7) + ';' +
REAL_TO_STRING(ptrAI^.hodnotaK8) + '$r$n';
            END_CASE;
            s := LEN(IN := sline);
            Memcpy(source := void(sline), dest := void(bufferTXT[j]), length := s);
            j := j + s;
          END_FOR;
          savetofile := true;
        ELSE
          busy := false;
          done := true;
          j := 0;
        END_IF;
      END_IF;
    END_IF;
END_FUNCTION_BLOCK
```

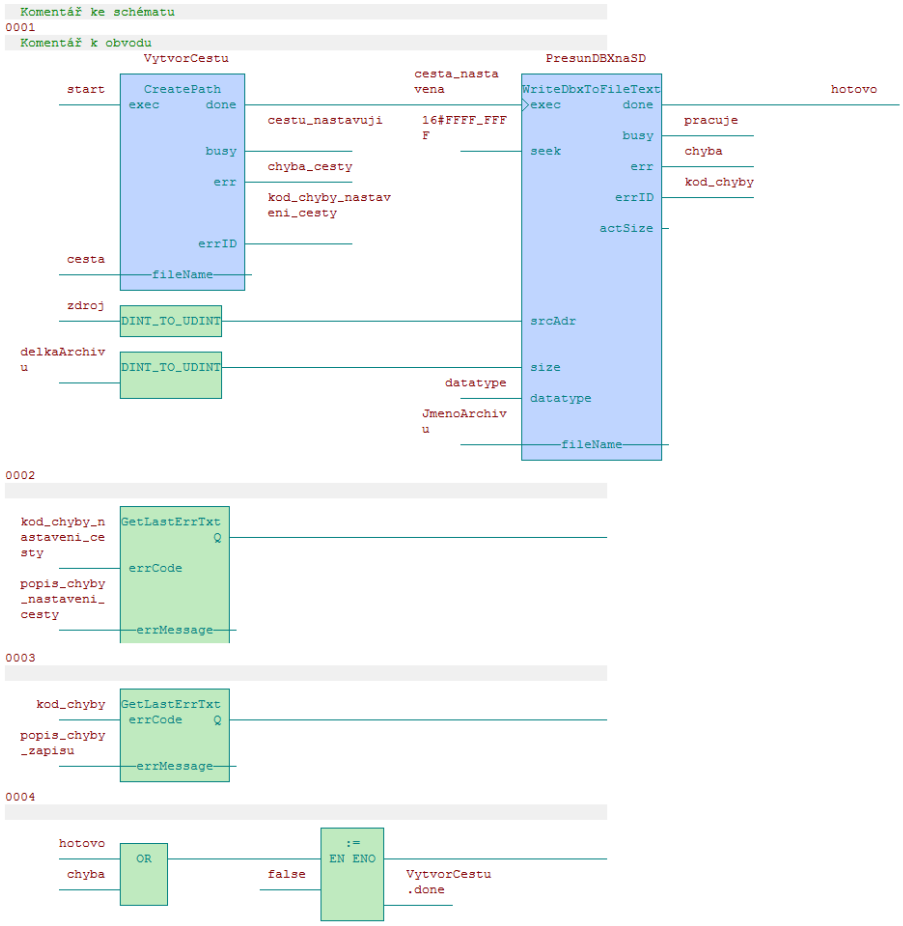
C:\MosaicApp\DiplomovaPrace\logger2\prgTeplyRestart.st

```
PROGRAM prgTeplyRestart
(* kod v tomto programu se provede po teplém restartu PLC.louží k inicializaci proměnných *)
  VAR_INPUT
  END_VAR
  VAR
  END_VAR
  VAR_OUTPUT
  END_VAR
  VAR_IN_OUT
  END_VAR
  VAR_EXTERNAL
  END_VAR
  VAR_TEMP
  END_VAR
  citAA := r0_p3_CNT_IN2.VALA;
  citBB := r0_p3_CNT_IN2.VALB;
 END_PROGRAM
```

C:\MosaicApp\DiplomovaPrace\logger2\fbZapisNaSD.FBD



C:\MosaicApp\DiplomovaPrace\logger2\prgSms.FBD



Příloha 2- Export zdrojového kódu z C#

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Globalization;
using System.IO;
using Excel = Microsoft.Office.Interop.Excel;
namespace CSVDataVisualiser
{
  public partial class Form1 : Form
  { char separatorDat = ';';
    char separatorDesetinychMist = '.';
    string JPG_PATH = Path.Combine(Path.GetTempPath(), "CSVDataVisualiser");
    int pocitadlo = 0;
    TypDat typDat = TypDat.Sms;
    List<PrvekAI> dataAI = new List<PrvekAI>();
    List<PrvekC> dataC = new List<PrvekC>();
    List<PrvekD> dataD = new List<PrvekD>();
    List<PrvekSms> dataSms = new List<PrvekSms>();
    NumberFormatInfo provider = new NumberFormatInfo();
   #region Methods
    public Form1()
```

```
    {
      InitializeComponent();
      provider.NumberDecimalSeparator = separatorDesetinychMist.ToString();
    }
    #region On Events
   protected override void OnHandleDestroyed(EventArgs e)
    {
      base.OnHandleDestroyed(e);
      pictureBox1.Image = null;
      for (int i = 0; i <= pocitadlo; i++)
        if (File.Exists(JPG_PATH + pocitadlo.ToString()))
          try
          {
            File.Delete(JPG_PATH + pocitadlo.ToString());
          }
          catch
          {
          }
    }
    private void btnFile_Click(object sender, EventArgs e)
    {
      OpenFileDialog openFileDialog = new OpenFileDialog();
      openFileDialog.Filter = "CSV files (*.CSV)|*.CSV|TXT files (*.TXT)|*.TXT";
      openFileDialog.RestoreDirectory = true;

      if (openFileDialog.ShowDialog() == DialogResult.OK)
      {
        try
        {
          SetEnableGui(false);
          NactiSoubor(openFileDialog.FileName);
          tbxFile.Text += openFileDialog.FileName + Environment.NewLine;
          SetEnableGui(true);
        }
        catch
        {
          MessageBox.Show(String.Format("Nepodařilo se načíst soubor:{0}{0}{1}", Environment.NewLine,
openFileDialog.FileName), "Chyba", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
      }
    }
    private void btnErase_Click(object sender, EventArgs e)
    {
      tbxFile.Text = String.Empty;
      dataAI = new List<PrvekAI>();
      dataC = new List<PrvekC>();
      dataD = new List<PrvekD>();
      dataSms = new List<PrvekSms>();
      dataGridView1.DataSource = null;
      pictureBox1.Image = null;
    }
   private void rbtAi_CheckedChanged(object sender, EventArgs e)
    {
      if (rbtAi.Checked)
        rbt_CheckedChanged(TypDat.AI);
    }
    private void rbtC_CheckedChanged(object sender, EventArgs e)
    {
      if (rbtC.Checked)
        rbt_CheckedChanged(TypDat.C);
    }
    private void rbtD_CheckedChanged(object sender, EventArgs e)
    {
      if (rbtD.Checked)
        rbt_CheckedChanged(TypDat.D);
    }
    private void rbtSms_CheckedChanged(object sender, EventArgs e)
    {
```

```
      if (rbtSms.Checked)
        rbt_CheckedChanged(TypDat.Sms);
    }
    private void pictureBox1_Click(object sender, EventArgs e)
    {
      if (pictureBox1.Image != null)
      {
        SaveFileDialog saveFileDialog = new SaveFileDialog();
        saveFileDialog.Filter = "JPG files (*.jpg)|*.jpg";
        saveFileDialog.RestoreDirectory = true;
        if (saveFileDialog.ShowDialog() == DialogResult.OK)
          pictureBox1.Image.Save(saveFileDialog.FileName);
      }
    }
    #endregion // On Events
    private void SetEnableGui(bool enable)
    {
      btnFile.Enabled = btnErase.Enabled = tbxFile.Enabled = rbtAi.Enabled = rbtC.Enabled =
rbtC.Enabled = rbtD.Enabled = rbtSms.Enabled = enable;
    }
    private void rbt_CheckedChanged(TypDat typ)
    {
      SetEnableGui(false);
      typDat = typ;
      if (!String.IsNullOrEmpty(tbxFile.Text))
        NactiSoubor(tbxFile.Text);
      SetEnableGui(true);
    }
    private void NactiSoubor(string filePath)
    {
      try
      {
        switch (typDat)
        {
          case TypDat.AI:
            foreach (string soubor in filePath.Split(new string[] { Environment.NewLine },
StringSplitOptions.RemoveEmptyEntries))
              if (!ZpracujSouborProAI(File.ReadAllLines(soubor).ToList()))
              {
                dataGridView1.DataSource = null;
                pictureBox1.Image = null;
                return;
              }
            VyplnDataGridView(dataAI.OrderBy(prvek => prvek.Date.Ticks).ToArray());
            break;

          case TypDat.C:
            foreach (string soubor in filePath.Split(new string[] { Environment.NewLine },
StringSplitOptions.RemoveEmptyEntries))
              if (!ZpracujSouborProC(File.ReadAllLines(soubor).ToList()))
              {
                dataGridView1.DataSource = null;
                pictureBox1.Image = null;
                return;
              }
            VyplnDataGridView(dataC.OrderBy(prvek => prvek.Date.Ticks).ToArray());
            break;

          case TypDat.D:
            foreach (string soubor in filePath.Split(new string[] { Environment.NewLine },
StringSplitOptions.RemoveEmptyEntries))
              if (!ZpracujSouborProD(File.ReadAllLines(soubor).ToList()))
              {
                dataGridView1.DataSource = null;
                pictureBox1.Image = null;
                return;
              }
            VyplnDataGridView(dataD.OrderBy(prvek => prvek.Date.Ticks).ToArray());
```

```
            break;
          case TypDat.Sms:
            foreach (string soubor in filePath.Split(new string[] { Environment.NewLine },
StringSplitOptions.RemoveEmptyEntries))
              if (!ZpracujSouborProSms(File.ReadAllLines(soubor).ToList()))
              {
                dataGridView1.DataSource = null;
                pictureBox1.Image = null;
                return;
              }
            VyplnDataGridView(dataSms.OrderBy(prvek => prvek.Date.Ticks).ToArray());
            break;
          default:
            break;
        }
      }
     catch
      {
        MessageBox.Show(String.Format("Nepodařilo se načíst soubor:{0}{0}{1}", Environment.NewLine,
tbxFile.Text), "Chyba", MessageBoxButtons.OK, MessageBoxIcon.Error);
        dataGridView1.DataSource = null;
      }
      try
      {   NakresliGraf();
      }
      catch
      {
        pictureBox1.Image = null;
        MessageBox.Show("Nepodařilo se vytvořit graf.", "Chyba", MessageBoxButtons.OK,
MessageBoxIcon.Error);
      }
    }
    #region Zpracování souboru
    private bool ZpracujSouborProAI(List<string> allLines)
    {
      try
      { dataAI.AddRange((from line in allLines
                         let d = line.Split(separatorDat)
                         select new PrvekAI(PrevedNaDatumACas(Convert.ToString(d[0])),
                                        Convert.ToDouble(d[1], provider),
                                        Convert.ToDouble(d[2], provider),
                                        Convert.ToDouble(d[3], provider),
                                        Convert.ToDouble(d[4], provider))).ToArray());
      }
      catch
      {
        MessageBox.Show("Načtený soubor neobasahuje informace AI ve správném formátu.", "Chyba",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return false;
      }
      return true;
    }
    private bool ZpracujSouborProC(List<string> allLines)
    {
      try
      {
        dataC.AddRange((from line in allLines
                        let d = line.Split(separatorDat)
                        select new PrvekC(PrevedNaDatumACas(Convert.ToString(d[0])),
                                       Convert.ToDouble(d[1], provider),
                                       Convert.ToDouble(d[2], provider))).ToArray());
      }
      catch
      {
        MessageBox.Show("Načtený soubor neobasahuje informace C ve správném formátu.", "Chyba",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return false;
      }
```

```
      return true;
    }
    private bool ZpracujSouborProD(List<string> allLines)
    {
      try
      {  dataD.AddRange((from line in allLines
                         let d = line.Split(separatorDat)
                         select new PrvekD(PrevedNaDatumACas(Convert.ToString(d[0])),
                                           Convert.ToDouble(d[1], provider),
                                           Convert.ToDouble(d[2], provider))).ToArray());
      }
      catch
      {
        MessageBox.Show("Načtený soubor neobasahuje informace D ve správném formátu.", "Chyba",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return false;
      }
      return true;
    }
    private bool ZpracujSouborProSms(List<string> allLines)
    {
      List<string> textSms = (from line in allLines
       let s = line.Split(new string[] { String.Concat(separatorDat,"\"") }, StringSplitOptions.None)
       select s.ElementAt(s.Length - 1).Substring(0, s.ElementAt(s.Length - 1).Length - 1)).ToList();
      try
      {
        dataSms.AddRange((from line in textSms
                          let d = line.Split(separatorDat)
                          select new PrvekSms(PrevedNaDatumACas(Convert.ToString(d[0])),
                                           Convert.ToDouble(d[1], provider),
                                           Convert.ToDouble(d[2], provider),
                                           Convert.ToDouble(d[3], provider),
                                           Convert.ToDouble(d[4], provider),
                                           Convert.ToDouble(d[5], provider),
                                           Convert.ToDouble(d[6], provider),
                                           Convert.ToDouble(d[7], provider),
                                           Convert.ToDouble(d[8], provider))).ToArray());
      }
      catch
      {
        MessageBox.Show("Načtený soubor neobasahuje informace ze SMS ve správném formátu.", "Chyba",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return false;
      }
      return true;
    }
    private DateTime PrevedNaDatumACas(string datumACas)
    {
      CultureInfo culture = new CultureInfo("ko-KR");
      DateTimeStyles styles = DateTimeStyles.None;
     string prvniCast = datumACas.Substring(0, 10);
      string druhaCast = datumACas.Substring(11, 8);
      DateTime dc = new DateTime();
      DateTime.TryParse(String.Concat(prvniCast, " ", druhaCast), culture, styles, out dc);
      return dc;
    }
    #endregion // Zpracování souboru
    private void VyplnDataGridView(object o)
    {
      dataGridView1.DataSource = o;
    }
    #region Excel
    private void NakresliGraf()
    {
      Excel.Application xlApp;
      Excel.Workbook xlWorkBook;
      Excel.Worksheet xlWorkSheet;
      object misValue = System.Reflection.Missing.Value;
```

```
  xlApp = new Excel.ApplicationClass();
  xlWorkBook = xlApp.Workbooks.Add(misValue);
  xlWorkSheet = (Excel.Worksheet)xlWorkBook.Worksheets.get_Item(1);
  Excel.Range chartRange = xlWorkSheet.get_Range("A1", "A1");
  switch (typDat)
  {
    case TypDat.AI:
      VlozHodnotyAIDoExcelu(ref xlWorkSheet);
      chartRange = xlWorkSheet.get_Range("A1", String.Concat("E", dataAI.Count + 1));
      break;
    case TypDat.C:
      VlozHodnotyCDoExcelu(ref xlWorkSheet);
      chartRange = xlWorkSheet.get_Range("A1", String.Concat("C", dataC.Count + 1));
      break;
    case TypDat.D:
      VlozHodnotyDDoExcelu(ref xlWorkSheet);
      chartRange = xlWorkSheet.get_Range("A1", String.Concat("C", dataD.Count + 1));
      break;
    case TypDat.Sms:
      VlozHodnotySmsDoExcelu(ref xlWorkSheet);
      chartRange = xlWorkSheet.get_Range("A1", String.Concat("I", dataSms.Count + 1));
      break;
    default:
      return;
  }
 Excel.ChartObjects xlCharts = (Excel.ChartObjects)xlWorkSheet.ChartObjects(Type.Missing);
  Excel.ChartObject myChart = (Excel.ChartObject)xlCharts.Add(10, 80, 600, 300);
  Excel.Chart chartPage = myChart.Chart;
 chartPage.SetSourceData(chartRange, Excel.XlRowCol.xlColumns);
  chartPage.ChartType = Excel.XlChartType.xlLine;
  chartPage.Export(JPG_PATH + pocitadlo.ToString() + ".jpg", "jpg", misValue);
  pictureBox1.Image = new Bitmap(JPG_PATH + pocitadlo.ToString() + ".jpg");
  xlWorkBook.Close(false, misValue, misValue);
  xlApp.Quit();
  releaseObject(xlWorkSheet);
  releaseObject(xlWorkBook);
  releaseObject(xlApp);
  pocitadlo++;
}
private void releaseObject(object obj)
{
  try
  { System.Runtime.InteropServices.Marshal.ReleaseComObject(obj);
    obj = null;
  }
  catch (Exception ex)
  {
    obj = null;
    MessageBox.Show("Exception Occured while releasing object " + ex.ToString());
  }
  finally
  {
    GC.Collect();
  }
}
private void VlozHodnotyAIDoExcelu(ref Excel.Worksheet xlWorkSheet)
{ xlWorkSheet.Cells[1, 1] = "Datum";
  for (int i = 2; i < 6; i++ )
    xlWorkSheet.Cells[1, i] = String.Concat("AI", i - 1);
  int radek = 2;
  foreach (PrvekAI prvekAI in dataAI.OrderBy(prvek => prvek.Date.Ticks))
  {
    xlWorkSheet.Cells[radek, 1] = prvekAI.Date.ToString("yyyy-MM-dd-HH:mm:ss");
    xlWorkSheet.Cells[radek, 2] = prvekAI.AI1;
    xlWorkSheet.Cells[radek, 3] = prvekAI.AI2;
    xlWorkSheet.Cells[radek, 4] = prvekAI.AI3;
    xlWorkSheet.Cells[radek, 5] = prvekAI.AI4;
    radek++;
```

```
      }
    }
    private void VlozHodnotyCDoExcelu(ref Excel.Worksheet xlWorkSheet)
    {      xlWorkSheet.Cells[1, 1] = "Datum";
     for (int i = 2; i < 4; i++)
        xlWorkSheet.Cells[1, i] = String.Concat("C", i - 1);
      int radek = 2;
      foreach (PrvekC prvekC in dataC.OrderBy(prvek => prvek.Date.Ticks))
      {
        xlWorkSheet.Cells[radek, 1] = prvekC.Date.ToString("yyyy-MM-dd-HH:mm:ss");
        xlWorkSheet.Cells[radek, 2] = prvekC.C1;
        xlWorkSheet.Cells[radek, 3] = prvekC.C2;
        radek++;
      }
    }
    private void VlozHodnotyDDoExcelu(ref Excel.Worksheet xlWorkSheet)
    { xlWorkSheet.Cells[1, 1] = "Datum";
      for (int i = 2; i < 4; i++)
        xlWorkSheet.Cells[1, i] = String.Concat("D", i - 1);
      int radek = 2;
      foreach (PrvekD prvekD in dataD.OrderBy(prvek => prvek.Date.Ticks))
      {        xlWorkSheet.Cells[radek, 1] = prvekD.Date.ToString("yyyy-MM-dd-HH:mm:ss");
        xlWorkSheet.Cells[radek, 2] = prvekD.D1;
        xlWorkSheet.Cells[radek, 3] = prvekD.D2;
        radek++;
      }
    }
    private void VlozHodnotySmsDoExcelu(ref Excel.Worksheet xlWorkSheet)
    {      xlWorkSheet.Cells[1, 1] = "Datum";
      for (int i = 2; i < 4; i++)
        xlWorkSheet.Cells[1, i] = String.Concat("D", i - 1);
      for (int i = 4; i < 6; i++)
        xlWorkSheet.Cells[1, i] = String.Concat("C", i - 3);
      for (int i = 6; i < 10; i++)
        xlWorkSheet.Cells[1, i] = String.Concat("AI", i - 5);
      int radek = 2;
      foreach (PrvekSms prvekSms in dataSms.OrderBy(prvek => prvek.Date.Ticks))
      {
        xlWorkSheet.Cells[radek, 1] = prvekSms.Date.ToString("yyyy-MM-dd-HH:mm:ss");
        xlWorkSheet.Cells[radek, 2] = prvekSms.D1;
        xlWorkSheet.Cells[radek, 3] = prvekSms.D2;
        xlWorkSheet.Cells[radek, 4] = prvekSms.C1;
        xlWorkSheet.Cells[radek, 5] = prvekSms.C2;
        xlWorkSheet.Cells[radek, 6] = prvekSms.AI1;
        xlWorkSheet.Cells[radek, 7] = prvekSms.AI2;
        xlWorkSheet.Cells[radek, 8] = prvekSms.AI3;
        xlWorkSheet.Cells[radek, 9] = prvekSms.AI4;
        radek++;
      }
    }
    #endregion // Excel
    #endregion // Methods
    enum TypDat
    {
      AI,
      C,
      D,
      Sms
    }
    private void button1_Click(object sender, EventArgs e)
    {
        this.Close();
    }
    private void groupBox1_Enter(object sender, EventArgs e)
    {
    }
  }
}
```

Příloha 3- Obsah a struktura CD

Diplomová práce

- *Diplomová_práce_Stařík.docx*
- *Diplomová_práce_Stařík.pdf*

Aplikační SW pro PLC

- *_PG_DiplomovaPrace_2011-05-07.piz*

Vizualizace v C#

- *VizualizaceDat.exe*
- *VizualizaceDat.rar*

Manuály

- *Knihovna FileLib pro práci se soubor.pdf*
- *Knihovna funkcí.pdf*
- *Knihovna pro GSM bránu.pdf*
- *Periferní moduly PLC Tecomat Foxtrot.pdf*
- *Práce s paměťovou kartou.pdf*
- *Programování PLC TECOMAT.pdf*
- *Programovatelné automaty Tecomat Foxtrot.pdf*
- *Příručka programátora PLC TECOMAT.pdf*
- *Příručka projektanta systémů Foxtrot.pdf*
- *TC_C35i_Terminal Siemens.pdf*
- *Začínáme v prostředí MOSAIC.pdf*